AD-A102 278 GENERAL DYNAMICS FORT WORTH TEX CONVAIR AEROSPACE DIV F/6 17/9
ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING. VOLUME II. PAR--ETC(U)
JUN 76 R J HANCOCK, F H CLEVELAND F30602-73-C-0380
UNCLASSIFIED RADC-TR-76-186-VOL-2-PT-1 NL

1 of 4
AD
A102278

LEVEL III 2

Vol 3 - A030572
V1-Pt1-A030555

RADC-TR-76-186, Vol II, Pt 1
Final Technical Report
June 1976

ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING
(Computer Program Documentation)

General Dynamic

Approved for public release;
distribution unlimited.

Laboratory Directors' Fund Number 01707310
with Proj. 6512 funds added.

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

APPROVED: John C. Cleary
JOHN C. CLEARY
Project Engineer

APPROVED: Moses A. Diab
MOSES A. DIAB
Acting Technical Director
Surveillance Division

FOR THE COMMANDER: John P. Huss
JOHN P. HUSS
Acting Chief, Plans Office

Initially funded under RADC Laboratory Directors' Funds with Proj. 6512 funds added.

Do not return this copy; retain or destroy.

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

19 TR-76-186-VOL-2-PT-1

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
12 RADC-TR-76-186 Vol II, Pt 1

2. GOVT ACCESSION NO.
AD-A102 278

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
6 ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING Computer Program Documentation. Volume II, Part I.

5. TYPE OF REPORT & PERIOD COVERED
9 Final Technical Report,

6. PERFORMING ORG. REPORT NUMBER
N/A

7. AUTHOR(s)
10 Robert J./Hancock
Fred H./Cleveland

8. CONTRACT OR GRANT NUMBER(s)
15 F30602-73-C-0380

9. PERFORMING ORGANIZATION NAME AND ADDRESS
General Dynamics
Convair Aerospace Division
P O Box 748, Ft Worth TX 76101

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
61101F
16 LDF 01707310 17 73

11. CONTROLLING OFFICE NAME AND ADDRESS
11 Rome Air Development Center (OCSA)
Griffiss AFB NY 13441

12. REPORT DATE
June 1976

13. NUMBER OF PAGES
380 12 344

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Same

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: John C. Cleary (OCSA)
Initially funded under RADC Laboratory Directors' Funds with Proj. 6512 funds added.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Radar
Simulation
Computer Modeling

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This effort is concerned with the development and implementation of a set of dital computer programs that will augment the RADC digital computer radar simulation model procured under Contr F30602-72-C-0393 (01707201). The computer programs shall consist of a sequence of subroutines that correspond to separate functions such as a chaff model, target model, propagation effects and clutter model. The original radar simulation model will be expanded to include a bistatic capability and will include ECM and phase coded pulse compression re-

ceiver techniques. In addition, an interactive system has been designed for the simulation. Using an interactive display, an engineer would be able to understand what is happening by being able to observe results at several intermediate points in the problem. A picture is worth a thousand words. For example, an antenna pattern or waveform response to a target is more meaningful than a long table of numerical listings. Parts of the simulation were used by RADC for Deep Space Surveillance Radar (DSSR) waveform analysis, generating antenna patterns and tradeoffs involving phase shifter bit-size for the Advanced Space Defense Program (ASDP). The RADC radar simulation model is being used to support Seek Sail, Cobra Judy, Digital Coded Radar and Seek Sentry.

This report contains a large percentage of machine-produced copy which is not of the highest printing quality but because of economical consideration, it was determined in the best interest of the government that they be used in this publication.

This report contains Vol I, Pt 1 (Sections 1-7 and 9-10) (Pages 1-1 thru 1-5, 2-1 thru 2-24, 3-1 thru 3-35, 4-1 thru 4-23, 5-1 thru 5-6, 6-1 thru 6-39, 7-1 thru 7-30, 9-1 thru 9-3 and 10-1 thru 10-2).

Vol I, Pt 2 contains Section 8 (Pages 8-1 thru 8-174).

Vol I, Pt 3 contains Section 8 (Pages 8-175 thru 8-418).

Vol II, Pt 1 contains Sections 1 - 8 and 10 & 11 (Pages 1-1, 2-1 thru 2-24, 3-1 thru 3-15, 4-1 thru 4-137, 5-1 thru 5-16, 6-1 thru 6-44, 7-1, 8-1 thru 8-26, 10-1 thru 10-4 and 11-1 thru 11-2).

Vol II, Pt 2 contains Sections 9 and 10 (Pages 9-1 thru 9-234 and Pages 10-1 thru 10-4).

Vol III contains Sections 1 thru 6 (Pages 1-1 thru 1-2, 2-1 thru 2-22, 3-1 thru 3-53, 4-1 thru 4-141, 5-1 thru 5-3 and 6-1).

Vol IV, Pt 1 contains Appendices A-K and Appendix M.

Vol IV, Pt 2 contains Appendix L.

TABLE OF CONTENTS

LIST OF FIGURES

SECTION 1

INTRODUCTION

This volume of the final technical report contains the descriptions of the computer programs and subprograms which constitute the Radar System Simulation Model.

The simulation model computer program is divided into two distinct phases: (1) Data initialization acivity, and (2) Simulation activity. In terms of software, each phase is composed of one main and a group of subprograms.

The subprograms are divided into five distinct groups: Stimulus/transfer function modules, connection modules, peripheral modules, supervisory modules and subordinate modules. These groups are more fully described in the sections devoted to each group.

The flow charts and cross reference tables for the entire Radar System Simulation Model are located in Part 2 of this volume.

The module descriptions, program listings, flow charts and cross reference tables have all been cross indexed and are located in Section 10. This section has been included in both parts of the volume for convenience.

SECTION 2

SIMULATION DATA LOADER EXECUTIVE

(MAIN-1)

The simulation data loader serves as the interface between the user and the segment of the computer program which performs the simulation. This segmentation was necessary since input data in a format convenient to the user must be converted to a form suitable for use by the simulation modules. The punch cards which define the simulation to be performed are of two types: simulation control cards and module parameter data cards. The simulation control cards determine what operations are to be performed in the simulation activity. This includes not only the scheduling of modules for execution but also the movement of data to and from temporary storage, and the modification of parameters for multiple executions of a simulation model configuration. Each control card with the exception of ENDPAS, ENDCFG, and ENDJ0B is converted into a control word by the control word generator and placed in the control word block. Each control word contains the code number of the operation to be performed, the module to be executed and data set reference number if required. The module parameter data cards define the parameters of the simulation modules to be executed in the simulation. Each module requiring input data has a unique namelist which contains the input parameters.

Figure 2-1 is a block diagram of the Initializer load module. In the block diagram the data flow paths are shown as solid lines and control paths are shown as dashed lines. Control normally is retained by the Simulation Data Loader Executive but control transfer does occur to the subprograms CLINT and PHENC. The blocks containing the letter M represent data transfers between two storage areas. Arrows are used to indicate the direction of data flow. The blocks containing C&M perform perform a search operation to determine if the word stored in a buffer is a member of a reference dictionary. If the search is successful, the data in the buffer is transferred to the storage area designated by an arrow. The function served by certain blocks in the diagram are evident from their titles. Those blocks requiring further explanation are discussed in the following paragraphs.

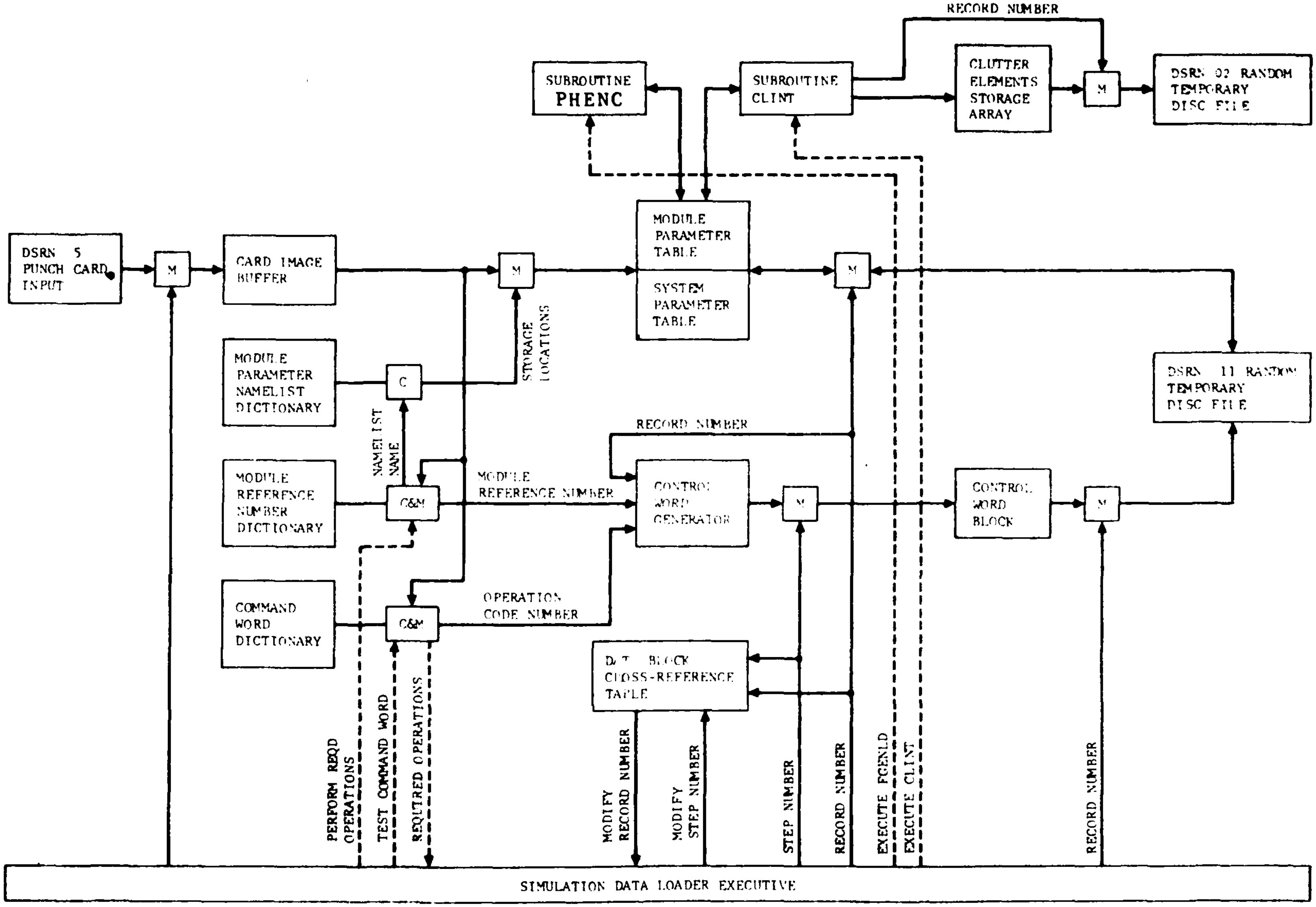


Figure 2-1 DATA INITIALIZATION LOAD MODULE FUNCTIONAL BLOCK DIAGRAM

The Module Parameter Namelist Dictionary serves the function of directing the input data to the proper location within the Module/System Parameter Table. Appendix B contains a list of the parameters in each namelist and the storage location assigned to each parameter. The namelist name is the same as the Module Reference Number, i.e., the namelist for module 101 is NL101.

The Module Reference Number Dictionary is a cross reference between Module Reference Numbers and the overhead operations which must be accomplished prior to execution of a module. For example, to generate a phase encoded waveform requires that the user supplied data be preprocessed by subroutine PHENC.

The Command Word Dictionary is a cross reference between input command words and the operations to be performed in the simulation. The control word is entered in columns 1 through 6 of the simulation control card. The following is a list of the command words and the corresponding operation initiated by each:

EXEC — This command word schedules execution of the module corresponding to the number appearing in columns 15 through 17 of the control card. No input data cards are required.

LDEXEC — This command word is the same as EXEC with the additional requirement that input data is read. The namelist name for entering the data is the same as the module reference number contained in columns 15 through 17 of the simulation control card.

MØDIFY — This command word causes the data loaded in a previous step to be modified. This is typically used to change parameters when multiple simulation passes are to be made. The step number to be changed is entered in columns 10 through 12 of the simulation control card, right adjusted. The Module Reference Number is entered in columns 15 through 17 of the simulation control card.

ENDPAS — This command word signifies the end of a pass through a simulation configuration.

ENDCFG This command word signifies the end of a configuration.

ENDJΘB This command word signifies the end of a simulation job.

STΘREX This command word causes the contents of the XT signal storage array to be stored on a temporary data set, usually a disc file. The Data Set Reference Number is entered in columns 11 and 12 of the simulation control card.

STΘREY Same as STΘREX except the contents of signal storage array YT are stored.

STΘREA Same as STΘREX except the contents of auxiliary storage array XA are stored.

STΘREB Same as STOREX except the contents of auxiliary storage array XB are stored.

LΘAD X This command word causes the data located in a temporary file to be loaded into the signal storage array XT. The Data Set Reference Number is entered in columns 11 and 12 of the simulation control card.

LΘAD Y Same as LΘAD X except the signal storage array YT is loaded.

LΘAD A Same as LΘAD X except the auxiliary storage array XA is loaded.

LΘAD B Same as LΘAD X except the auxiliary storage array XB is loaded.

REWIND This command word causes a temporary file to be rewound. For a disc file this moves the access pointer to the beginning of the data set. The Data Set Reference Number is entered in columns 11 and 12 of the simulation control card.

The output from the data loader are stored on temporary storage files. Data Set Reference Number 11 contains the control word block which controls the simulation activity and the module/system parameter tables used to define the modules and system characteristics. Data Set Reference Number 2 contains the clutter scatterer parameters.

Flow Chart: Page 9-2

Cross Reference Table: Page 9-206

```
CARD NO.       ****                     CONTENTS                              ****

                  COMMON/BLK1/ CSCAT(3000)
                  COMMON /BLK2/ ITEMP(500),IEXEC(500)
                  COMMON/BLKRND/ XRNDM(141)
           C
           C
                  DIMENSION ANTAZ(2,75)  , ANTEL(2,75)
                  DIMENSION WTX(3,100) , WT(3,100) , FZERO(2,50) , FPOLE(2,50),      010
                 *GRAND(1,9),TEN(2,50)
           C
                  DIMENSION FCOM(300), IEC(3,32),IPY(8)
                  DIMENSION ISCAT(3,100)
                  DIMENSION ISPECS(100)
                  DIMENSION DFTIN(3,100)
                  DIMENSION ITAP(3,100)
                  DIMENSION FF1(11),FSAM(2,100),FBCOEF(2,25),FFCOEF(2,25)
           C
                   CHARACTER VIL*100,VCL*100,GENE*100
                   INTEGER  EXEC, ENDPAS, ENDJOB, ENDCFG, STOREX, STOREY, STOREA,    025
                 * STOREC, REWIND, OUTTX, OUTTY, OUTTA, OUTTC
                   INTEGER CLEARA,CLEARF,CLEARX,CLEARY,BLANK,SKIP
                  REAL LAMBDA,JSTART,JVEL,JFREQ0,JFC
                   REAL JFMC,JRSIM,JMAZ,JHGT,JERF,JFMEW,JPW
           C
                  DATA EXEC/'EXEC  '/  , COEXEC/'COEXEC'/ ,  ENDPAS/'ENDPAS'/  ,    125
                 *    ENDJOB/'ENDJOB'/  , ENDCFG/'ENDCFG'/ ,  MODIFY/'MODIFY'/  ,    126
                 *    STOREX/'STOREX'/  , STOREY/'STOREY'/ ,  STOREA/'STOREA'/  ,    127
                 *    STOREC/'STOREC'/  , LOAD X/'LOAD X'/ ,  LOAD Y/'LOAD Y'/  ,    128
                      LOAD A/'LOAD A'/  , LOAD C/'LOAD C'/ ,  REWIND/'REWIND'/  ,    129
                 *   EDIT X/'EDIT X'/  , EDIT Y/'EDIT Y'/ ,  EDIT A/'EDIT A'/  ,     130   λ-7
                 *   EDIT B/'EDIT B'/  , INPUTX/'INPUTX'/ ,  INPUTY/'INPUTY'/  ,     131
                 *   INPUTA/'INPUTA'/  , INPUTB/'INPUTB'/ ,SKIP/'SKIP  '/
                 DATA NCLDEX/'NCLDEX'/, CLEARX/'CLEARX'/, CLEARY/'CLEARY'/,                360
                 *      CLEARA/'CLEARA'/, CLEARB/'CLEARB'/,BLANK/'      '/
           C
                 EQUIVALENCE    (ITEMP(  1), NZ          ),(ITEMP(  2), FS          ),  042
                 *              (ITEMP(  3), FFFO        ),(ITEMP(  4), SIMBW       ),  043
                 *              (ITEMP(  5), ISDUMP      ),(ITEMP(  6), ICINV       ),  044
                 *              (ITEMP(  7), ICFLF       ),(ITEMP(  8), SIMFO       ),  045
                 *              (ITEMP(  9), NORMFT      ),(ITEMP( 10), WSCAN       ),  046
                 *              (ITEMP( 11), FI          ),(ITEMP( 12), TI          ),  047
                 *              (ITEMP( 13), LAMBDA      ),(ITEMP( 14), RNGCEL      ),  048
                 *              (ITEMP( 15), FSHIFT      ),(ITEMP( 16), TIME        ),  049
                 *              (ITEMP( 17), AZANG       ),(ITEMP( 18), ANTAZO      ),  050
                 *              (ITEMP( 19), ELANG       ),(ITEMP( 20), ANTELO      ),  051
                 *              (ITEMP(150), AZBST       ),(ITEMP(151), NPTAZ       ),  052
                 *              (ITEMP(152), ELBST       ),(ITEMP(153), NPTEL       ),  053
                 *              (ITEMP(201), OFTIN(1,1)),(ITEMP(200), NIMP        ),  054
                 *              (ITEMP(201), ANTAZ(1,1)),(ITEMP(351), ANTEL(1,1))   055
                 * ,           (ITEMP(130), PRI(1)      ),(ITEMP(142), NPRIS       ),  056
                 *             (ITEMP(143), IPRIPT      ),(ITEMP(144), NREPET    )
           C
                 EQUIVALENCE    (ITEMP( 21), IDDUMP      ),(ITEMP( 22), IRND        ),  059
                 *              (ITEMP( 23), IADD1       ),(ITEMP( 24), JRND        ),  060
                 *              (ITEMP( 25), UMEAN       ),(ITEMP( 26), UEXT        ),  061
                 *              (ITEMP( 27), XMEAN       ),(ITEMP( 28), SIGMA       ),  062
                 *              (ITEMP( 29), UGMRND      ),(ITEMP( 30), SIG2SQ      ),  063
                 *              (ITEMP( 31), UL          ),(ITEMP( 32), XUEXT       ),  064
```

```
                  *          (ITEMP([illegible]), [illegible](1)   )                           065

              C

                EQUIVALENCE  (ITEMP( 33), ASELT     ),(ITEMP( 34), ASTAT      ),  067
                  *          (ITEMP( 35), ASTOP     )                              068
              C                                                                    069
                EQUIVALENCE  (ITEMP( 36), NAVL      ),(ITEMP( 37), NPWT       ),  070
                  *          (ITEMP( 38), EFIG      ),(ITEMP(101), WT(1,1)    )   071
              C                                                                    072
                EQUIVALENCE  (ITEMP( 39), TLIM      ),(ITEMP( 40), GLIM       ),  073
                  *          (ITEMP( 41), NHIST     ),(ITEMP( 42), NCPACK     ),  074
                  *          (ITEMP( 43), NLPACK    )                              075
              C                                                                    076
                EQUIVALENCE  (ITEMP( 44), NENOPT    ),(ITEMP( 45), NTYPER     )   077
                EQUIVALENCE  (ITEMP( 46), NRCS      ),(ITEMP( 47), NRWPR      ),  078
                  *          (ITEMP( 48), RWPR      ),(ITEMP( 49), WNDVEL     ),  079
                  *          (ITEMP( 50), VELANG    ),(ITEMP( 51), ENEXT      ),  080
                  *          (ITEMP( 52), RNG       ),(ITEMP( 53), AZEXT      ),  081
                  *          (ITEMP( 54), AZO       ),(ITEMP( 55), MM         ),  082
                  *          (ITEMP( 56), ELEXT     ),(ITEMP( 57), ELO        ),  083
                  *          (ITEMP( 58), NN        ),                             084
                  *          (ITEMP(120), KK        ),(ITEMP(121), NBLKS      ),  085
                  *          (ITEMP(122), MODE      ),(ITEMP(123), DELAZ      ),  086
                  *          (ITEMP(124), DELEL     ),(ITEMP(125), AVLANG     ),  087
                  *          (ITEMP(126), ICFLG     ),(ITEMP(127), DOPFRQ     )   088
                  *  ,      (ITEMP(128), KCELL     )                              089
              C                                                                    090
                EQUIVALENCE  (ITEMP(113), IFCODE    )                                   920
                EQUIVALENCE  (ITEMP( 59), ST        ),(ITEMP( 60), ENG        ),  091
                  *          (ITEMP( 61), EE        ),(ITEMP( 62), EF         ),  092
```

```
                    *        (ITEMP( 63), NSRF       ),(ITEMP( 64), NAGTE      ),  093  2-9
                    *        (ITEMP( 65), TP         ),(ITEMP( 66), TE         ),  094
                    *         (ITEMP(201),  VIL),(ITEMP(251),VDL),
                    *        (ITEMP(301), GLBL       )
              C                                                                    097
                EQUIVALENCE  (ITEMP( 67), NTEN       ),(ITEMP(201), IFN(1,1)  )    098
              C
                EQUIVALENCE  (ITEMP( 68), FF0        ),(ITEMP( 69), FF1        ),  100
                    *        (ITEMP( 70), FB1        ),(ITEMP( 71), FB2        )   101
                EQUIVALENCE (ITEMP( 68), FFF),(ITEMP( 69), FF1),                        1031
                    *       (ITEMP( 70), FBN),(ITEMP( 71), FB2)                         1032
                EQUIVALENCE  (ITEMP(160), IFF0N      ),(ITEMP(161), IFF0D      ),  102
                    *        (ITEMP(162), IFF1N      ),(ITEMP(163), IFF1D      ),  103
                    *        (ITEMP(164), IFB1N      ),(ITEMP(165), IFB1D      ),  104
                    *        (ITEMP(166), IFB2N      ),(ITEMP(167), IFB2D      ),  105
                    *        (ITEMP(168), MODEDF     ),(ITEMP(169), NBITUF     )   106
                EQUIVALENCE (ITEMP(160), IFFRN),(ITEMP(161), IFFRD),                    1081
                    *       (ITEMP(162), IFFIN),(ITEMP(163), IFFID),                    1082
                    *       (ITEMP(164), IFBRN),(ITEMP(165), IFBRD),                    1083
                    *       (ITEMP(166), IFBIN),(ITEMP(167), IFBID)                     1084
              C
                EQUIVALENCE  (ITEMP( 72), NZ         ),(ITEMP( 73), NP         ),  108
                    *        (ITEMP( 74), SF         ),(ITEMP(201), FZERO(1,1)),   109
                    *        (ITEMP(301), FPOLE(1,1))                              110
              C                                                                    111
                EQUIVALENCE  (ITEMP( 75), FBCK       )                             112
              C                                                                    113
                EQUIVALENCE  (ITEMP( 76), NTSAF      ),(ITEMP( 77), DX         ),  114
                    *        (ITEMP( 78), THETAS     ),(ITEMP( 79), NROWS      ),  115
```

2-10

```
                        CONTENTS                                      ****

      *          (ITEMP( 80), NFPSCC    ),(ITEMP( 81), IFPSCC     ),  116
      *          (ITEMP( 82), NFPSK     ),(ITEMP( 83), IFPSK      ),  117
      *          (ITEMP( 84), INCRM     ),(ITEMP( 85), NPULS      ),  118
      *          (ITEMP( 86), IDLIML    ),(ITEMP( 87), ANGLE      ),  119
      *          (ITEMP( 88), NCUPLS    ),(ITEMP( 89), TCGNCM     )   120
      * ,       (ITEMP(115), TIMLSB),(ITEMP(116), MCDISK)                 1221
      * ,       (ITEMP(117), KECDEL),(ITEMP(118), KECIRI),
      *         (ITEMP(119), THETAR)
      * ,        (ITEMP(401), TSRECS(1)  )                            121
       EQUIVALENCE (ITEMP(125), TOLMCN)
 C                                                                    122
       EQUIVALENCE   (ITEMP( 90), FW         ),(ITEMP( 91), FSTRT      ),  123
      *              (ITEMP( 92), CHIRP      ),(ITEMP( 93), FMCW       ),  124
      *              (ITEMP( 94), NEWTX      ),(ITEMP( 95), SPW        ),  125
      *              (ITEMP( 96), NSUBP      ),(ITEMP( 97), SWTIM      ),  126
      *              (ITEMP( 98), KISTIM     ),(ITEMP( 99), FALTIM     ),  127
      *              (ITEMP(100), TSTART     ),(ITEMP(201), PCCODE(1)  ),  300
      *              (ITEMP(201), WTX(1,1) ),(ITEMP(114), CPRI ),
      *                (ITEMP(129), VPEAK   ),
      *               (ITEMP(194),  ICCDE    )  ,  (ITEMP(133),MODEPH),
      *                (ITEMP(164),  NSR   ),  (ITEMP(165),  IPY(1))
 C                                                                    130
       EQUIVALENCE   (ITEMP(101), NTAPS      ),(ITEMP(102), FCDEC      ),  131
      *              (ITEMP(154),  TAPSPC),  (ITEMP(155),  INFTF),
      *             (ITEMP(201), ITAP(1,1)  ),(ITEMP(201), DEC(1,1)  )   132
 C                                                                    133
       EQUIVALENCE   (ITEMP(103), XLSB       ),(ITEMP(104), NFITS      ),  134
      *             (ITEMP(105), INDFF      ),(ITEMP(149), ACCFS      )   135
 C                                                                    136
```

```
      EQUIVALENCE   (ITEMP(106), HTGT      ),(ITEMP(107), RTGT       ),  137
     *              (ITEMP(108), ANGTGT    ),(ITEMP(109), TCRINT     ),  138
     *              (ITEMP(110), RSIM      ),(ITEMP(111), NSCAT      ),  139
     *              (ITEMP(112), TGTVEL    ),(ITEMP(201), TSCAT(1,1))     140
      EQUIVALENCE  (ITEMP(170), NCELL      ),(ITEMP(198), TAVG       )
       EQUIVALENCE  (ITEMP(145), GAIN      )
      EQUIVALENCE  (ITEMP(180), SHTG       ),(ITEMP(181), SHPHAS     )    142
     * ,            (ITEMP(182), TJIT      )                                143
                                                                            144
C
       EQUIVALENCE   (ITEMP(146), CFREQ  ), (ITEMP(147), XWLENG   ),
     *          (ITEMP(148),SEDENS        )
       EQUIVALENCE (ITEMP(195), RADIUS),  (ITEMP(196), NSAM  ),
     *             (ITEMP(197), NDF2),  (ITEMP(201), FSAM(1,1))
       EQUIVALENCE  (ITEMP(199), NSEC  ), (ITEMP(201), FFCOEF(1,1)),
     *             (ITEMP(251), FFCOEF(1,1))
       EQUIVALENCE  (ITEMP(171), JRNG ),
     *             (ITEMP(172), JRSIM ) , (ITEMP(173), JMAZ ),
     *             (ITEMP(174), JHGT  ) , (ITEMP(175), JERP ),
     *             (ITEMP(176), JFMBW ) , (ITEMP(177), JPW  ),
     *             (ITEMP(178), JFO   ) , (ITEMP(179), JSTART),
     *             (ITEMP(156), JVEL  ) , (ITEMP(157), JPERCD)
C
       NAMELIST/NL101/IADD1,IRND,JRND,UMEAN,UEXT,SIGMA,XMEAN,NRAND,        146
     *  IDDUMP
       NAMELIST/NL102/ UMEAN,UEXT,SIGMA,XMEAN,IDDUMP
        NAMELIST/NL104/ NAVG
        NAMELIST/NL105/ NAVG
        NAMELIST/NL106/ NAVG
        NAMELIST/NL108/ NAVG
```

```
[illegible]        ****                       CONTENTS                                        ****

[illegible]        NAMELIST/NL199/ ADFLT,ASTFT,ASTLF
[illegible]        NAMELIST/NL201/ SIMBW,DFTIN,NIMF,ILLUMF
[illegible]        NAMELIST/NL202/ NZ,FSHIFT,ICINV,ICFOR,SIMBW,NORMFT                  150
[illegible]        NAMELIST/NL203/ NZ,FSHIFT,ICINV,ICFOR,SIMBW,NORMFT
[illegible]        NAMELIST/NL208/ILIM,FLIM,NHIST,NCPACK,NDPACK
[illegible]        NAMELIST/NL209/ILIM,FLIM,NHIST,NCPACK,NDPACK
[illegible]        NAMELIST/NL210/ NCPACK
[illegible]        NAMELIST/NL211/ NCPACK
[illegible]        NAMELIST/NL212/ NCPACK
[illegible]        NAMELIST/NL213/ NCPACK
[illegible]        NAMELIST/NL214/ NENDPT,NTYPEF
[illegible]        NAMELIST/NL215/ NENDPT,NTYPEF
[illegible]        NAMELIST/NL216/ XLSP,NPITS,IROFF,ALCFS
[illegible]        NAMELIST/NL217/ XLSP,NPITS,IROFF,ALCFS
[illegible]        NAMELIST/NL218/ XLSP,NPITS,IRUFF
[illegible]        NAMELIST/NL219/ XLSP,NPITS,IRUFF
[illegible]        NAMELIST/NL220/ XLSP,NPITS,IRUFF
[illegible]        NAMELIST/NL221/ NEWT,ORIG,WT
[illegible]        NAMELIST/NL222/ NEWT,ORIG,WT
[illegible]        NAMELIST/NL223/ NEWT,ORIG,WT
[illegible]        NAMELIST/NL224/ SHTO,SHPHAS,SIMFC
[illegible]        NAMELIST/NL225/ SHTO,SHPHAS,SIMFC
[illegible]        NAMELIST/NL226/ SHTO,SHPHAS,SIMFC
[illegible]        NAMELIST/NL227/ SHTO,SHPHAS,SIMFC
[illegible]        NAMELIST/NL229/ SHTO,SHPHAS,SIMFO,TJIT
[illegible]        NAMELIST/NL230/ SHTO,SHPHAS,SIMFO,TJIT
[illegible]        NAMELIST/NL231/ SHTO,SHPHAS,SIMFC,TJIT
[illegible]        NAMELIST/NL232/ SHTO,SHPHAS,SIMFC,TJIT
[illegible]        NAMELIST/NL233/ [illegible],FFFO,SIMBW,DFTIN,NIMF,ILLUMF
```

```
  NAMELIST/NL234/ NZ,SIMBW,DFTIN,NIMF,IDDUMP,SIMFO   2-13
  NAMELIST/NL235/ NTYPER
  NAMELIST/NL236/ NTYPER
  NAMELIST/NL237/ NTYPER
 NAMELIST/NL301/ SIMBW,NZ,FS,RFFO ,ISDUMP,ICINV,ICFOR,SIMFO ,WSCAN 176
*,NFEPLT,NGFMFT,TIME,ANTAZO,ANTELO,AZEST,ELEST,NPTAZ,NPTEL,            177
* ANTAZ,ANTEL,RNGCEL,TI,FI,LAMBDA,NPRIS,PRI,IPRIPI                    178
 NAMELIST/NL302/ NECS,NRWPH ,RWPH,WNDVEL,VELANG,RNEXT,RNO,AZEXT,      179
* AZO,MM,ELEXT,ELO,NN,RNGCEL,IDDUMP
* ,KK,NELKS,MODE,DELAZ,DELEL,ICFLG,DOPPRC,KCELL
 NAMELIST/NL303/ ST,RNG,LF,VOL,NSKP,NAUTO,TH,TL,IFCCDC
 NAMELIST/NL304/ ST,RNG,LF,VOL,NSKP,NAUTO,TH,TL,IFCCDC
 NAMELIST/NL305/ ST,RNG,LF,VOL,NSKP,NAUTO,TH,TL,IFCCDC
 NAMELIST/NL306/ ST,RNG,LF,VOL,NSKP,NAUTO,TH,TL,IFCCDC
 NAMELIST/NL307/ ST,RNG,LF,VIL,VOL,GLBL,NAUTO,TH,TL,IFCCDC,NSKP
 NAMELIST/NL308/ ST,RNG,LF,VIL,VOL,GLBL,NAUTO,TH,TL,IFCCDC,NSKP
 NAMELIST/NL309/ ST,RNG,LF,VIL,VOL,GLBL,NAUTO,TH,TL,IFCCDC,NSKP
 NAMELIST/NL310/ ST,RNG,LF,VIL,VOL,GLBL,NAUTO,TH,TL,IFCCDC,NSKP
 NAMELIST/NL401/ TFN,NTFN
 NAMELIST/NL402/ TFN,NTFN
 NAMELIST/NL403/ FFR,FFI,FBR,FBI,IFFRN,IFFRD,IFFIN,IFFID,
* IFBRN,IFBRD,IFBIN,IFBID,NBITDF,MODEDF
 NAMELIST/NL404/ FFR,FFI,FBR,FBI,IFFRN,IFFRD,IFFIN,IFFID,
* IFBRN,IFBRD,IFBIN,IFBID,NBITDF,MODEDF
 NAMELIST/NL405/ FF0,FF1,FB1,FB2,IFF0N,IFF0D,IFF1N,IFF1D,IFB1N,     196
* IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
 NAMELIST/NL406/ FF0,FF1,FB1,FB2,IFF0N,IFF0D,IFF1N,IFF1D,IFB1N,     198
* IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
 NAMELIST/NL407/ NZ,NP,SF,FZERO,FPLL.
```

```
CARD NO        ****                         CONTENTS                          ****

               NAMELIST/NL408/ NZ,NF,SF,FZERO,FPOLE
               NAMELIST/NL409/ FECK
               NAMELIST/NL410/ FECK
               NAMELIST/NL415/ NTSAR,DX,REFC,THETAS,NROWS,NEPSCG,IRPSCG,NEPSR,    203
              * IRPSR,INCRM,IDDUMP,NPOLS,TSRECS
               NAMELIST/NL420/ PW,       FMEW,NPWTX,SPW,NSUBP,SWTIM,RISTIM,       205
              * FALTIM,TSTART,FCODE,WTX,IDDUMP,SIMFO,VPEAK,CHIRP,FSTRT
               NAMELIST/NL421/ PW,       FMEW,NPWTX,SPW,NSUBP,SWTIM,RISTIM,       207
              * FALTIM,TSTART,PCODE,WTX,IDDUMP,SIMFO,VPEAK,CHIRP,FSTRT
               NAMELIST/NL422/ NTAPS,DEC,FODEC,ITAP
               NAMELIST/NL423/ NTAPS,DEC,FODEC,ITAP
               NAMELIST/NL425/ DX,THETAS,NROWS,NEPSCG,IRPSCG,
              * NPOLS,MODPUL,ICONLM,NPWTX,FMEW,WTX,RISTIM,FALTIM,
              * IDDUMP,TIMESS
               NAMELIST/NL426/ NTSAR,DX,THETAS,NROWS,NEPSR,IRPSR,TCLINE,
              * ANGLE,TSRECS,IDDUMP,FOCDEL,THETAR,MODTSR,FECK,TCLMON
               NAMELIST/NL427/ NTSAR,DX,THETAS,NROWS,NEPSR,IRPSR,TCLINE,
              * ANGLE,IDDUMP,RECDEL,THETAR,MODTSR,FECK,TCLMON
               NAMELIST/NL430/ FFO,FF1,FB1,FB2,IFFON,IFFOD,IFF1N,IFF1D,IFB1N,    217
              * IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
               NAMELIST/NL431/ FFO,FF1,FB1,FB2,IFFON,IFFOD,IFF1N,IFF1D,IFB1N,    219
              * IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
               NAMELIST/NL432/ FFO,FF1,FB1,FB2,IFFON,IFFOD,IFF1N,IFF1D,IFB1N,    221
              * IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
               NAMELIST/NL433/ FFO,FF1,FB1,FB2,IFFON,IFFOD,IFF1N,IFF1D,IFB1N,    223
              * IFB1D,IFB2N,IFB2D,NBITDF,MODEDF
               NAMELIST/NL434/FECK
               NAMELIST/NL435/FECK
               NAMELIST/NL436/FECK
```

```
      NAMELIST/NL437/FBCK
      NAMELIST/NL440/ NCELL
      NAMELIST/NL441/ NCELL
      NAMELIST/NL451/ FBCK,FECDEL,RECIFT,THETAR,DX,NROWS
      NAMELIST/NL452/ FBCK,RECDEL,RECIRT,THETAR,DX,NROWS
      NAMELIST/NL453/ DX,THETAS,NROWS,NRPSCG,IRPSCG,
     * NPULS,NCGPLS,ICGNUM,NFWTX,FMBW,WTX,RISTIM,FALTIM,TIMLSB
       NAMELIST/NL454/ SIMFO,SPW,NSUBP,PPRI
       NAMELIST/NL455/ DX,THETAS,NROWS,ICGNUM,SPW
       NAMELIST/NL456/ GAIN
       NAMELIST/NL457/ GAIN
       NAMELIST/NL458/ GAIN
       NAMELIST/NL459/ TAVG
       NAMELIST/NL460/ TAVG
       NAMELIST/NL461/ NSEC,SF,FBCOEF,FFCOEF
       NAMELIST/NL462/ NSEC,SF,FBCOEF,FFCOEF
       NAMELIST/NL463/ RADIUS,NSAM,NDFZ,FSAM
      NAMELIST/NL501/ IDDUMP,HTGT,RTGT,ANGTGT,TCRINT,KSIM,NSCAT,
     * TSCAT,TGTVEL
      NAMELIST/NL502/ IDDUMP,HTGT,RTGT,ANGTGT,TCRINT,KSIM,NSCAT,
     * TSCAT,TGTVEL
      NAMELIST/NL504/ NTSAR,DX,RFFO,THETAS,NROWS,NRPSCG,IRPSCG,NBPSR,    235
     * IRPSR,INCFM,IDDUMP,NPULS,TSRECS
      NAMELIST/NL505/ IDDUMP,NTSAR,DX,THETAS,NROWS,NRPSR,IRPSR,TDLINE,   237
     * ADTRT,ADTOP,ADELT,TSRECS,FECDEL,THETAR,MODTSR,RECIFT,FBCK,
     * IDLMCN
       NAMELIST/NL506/ NSR,MODEPH,IPY,ICODE,CHIRP,FSTRT,SPW,NSUBP,SWTIM,
     * TSTART,SIMFO,VPEAK,RISTIM,FALTIM
       NAMELIST/NL507/ NSR,MODEPH,IPY,ICODE,CHIRP,FSTRT,SPW,NSUBP,SWTIM,
     * FSTART,SIMFC,VPEAK,RISTIM,FALTIM
      NAMELIST/NL506/ FODEC,TAPSPC,INPTF
      NAMELIST/NL509/ FODEC,TAPSPC,INPTF
      NAMELIST/NL510/ CFREC,XWLENG
      NAMELIST/NL511/ SIGENS
      NAMELIST/NL517/ JFNG,JFSIM,JMAZ,JHCT,JERR,JFMOW,JFW,
     *   JFC,JVEL,JFCFOD,RADIUS,NSAM,NDFZ,FSAM
      DATA N00001,N00045,N00141,N00455,N00500/1,45,141,455,500/        234
      DATA N00012,N00013,N00021,N00129/12,13,21,129/
C
C
      CALL FANSIZ(1,1,500)
      IEXEC=1
       IEXEC(500)=1
       ICLCK=1
      ICPU=1
       DO 2 J=1,8
       IFY(J)=1
    2  CONTINUE
      DO 3 J=1,2
      ITEMP(J)=0
    3 CONTINUE
    4 DO 4 J=1,50
      IEXEC(J) = 0
    4  CONTINUE
C
       IF(ICPU.EQ.1) WRITE(11*1) IEXEC
C
      DO 1010 I=1,500
```

```
 1614 ITEMP(I) = 0
      JSTEP=0
      ISIM=1
      IFLAGG = 1
       IFCFLG=0
       VPEAK=1.0
C
 2000 CONTINUE
      IF( IFLAGG .EQ. 0 ) GO TO 2003
      IFLAGG = 0
      WRITE(6,2001) ISIM,ICFG
 2001 FORMAT(1H1,'SIMULATION PASS',I4,'DATA LOAD FOR CONFIGURATION NUMBE273
     *R',I4,' IS BEGINNING')
 2003 CONTINUE
       FSHIFT=0.0
      IFC=0
      IGDUMP=0
       INFTF=0
      READ(5,8) ICARD,III,MODULE
    8 FORMAT(  A6    , 3X, I3, 3X, I3 )
       WRITE(6,18) ICARD,III,MODULE
   18 FORMAT(////,1H ,A6    ,10X,I3,10X,I3)
C
      IF( ICARD .EQ. NCLUDEX ) GO TO 1600
      IF( ICARD .EQ.    EXEC ) GO TO 1605
      IF( ICARD .EQ. LDEXEC ) GO TO 1610
      IF( ICARD .EQ. ENDPAS ) GO TO 1500
      IF( ICARD .EQ. ENDJOB ) GO TO 1800
      IF( ICARD .EQ. ENDCFG ) GO TO 1700
```

2-18

```
CARD NO          ****                     CONTENTS                               ****

                       IF( ICARD .EQ. MODIFY ) GO TO 1710
                       IF( ICARD .EQ.   SKIP ) GO TO 1710
                       JSTEP=JSTEP+1
                       IFC=111
                       IF( ICARD .EQ. STOREX ) GO TO 1615
                       IF( ICARD .EQ. STOREY ) GO TO 1620
                       IF( ICARD .EQ. STOREA ) GO TO 1625
                       IF( ICARD .EQ. STOREB ) GO TO 1630
                       IF( ICARD .EQ.  LOADX ) GO TO 1635
                       IF( ICARD .EQ.  LOADY ) GO TO 1640
                       IF( ICARD .EQ.  LOADA ) GO TO 1645
                       IF( ICARD .EQ.  LOADB ) GO TO 1650
                       IF( ICARD .EQ. REWIND ) GO TO 1655
                       IF( ICARD .EQ.  OUTTX ) GO TO 1660
                       IF( ICARD .EQ.  OUTTY ) GO TO 1665
                       IF( ICARD .EQ.  OUTTA ) GO TO 1670
                       IF( ICARD .EQ.  OUTTB ) GO TO 1675
                       IF( ICARD .EQ. INPUTX ) GO TO 1680
                       IF( ICARD .EQ. INPUTY ) GO TO 1685
                       IF( ICARD .EQ. INPUTA ) GO TO 1690
                       IF( ICARD .EQ. INPUTB ) GO TO 1695
                       IF( ICARD .EQ. CLEARX ) GO TO 1730
                       IF( ICARD .EQ. CLEARY ) GO TO 1735
                       IF( ICARD .EQ. CLEARA ) GO TO 1740
                       IF( ICARD .EQ. CLEARB ) GO TO 1745
                       JSTEP=JSTEP-1
                       WRITE(6,5)
                    5  FORMAT( ' INCORRECT ALPHA INPUT. CARD IGNORED. '/ )                     316
                       GO TO 1005
```

```
C
   31 CONTINUE
      IF(INDEX.EQ.415.OR.INDEX.EQ.426.OR.INDEX.EQ.504) GO TO 32
      IF(INDEX.EQ.505) GO TO 32
      GO TO 34
   32 DO 33 J=1,100
   33 TSKLCS(J)=1.0
   34 CONTINUE
       IF(INDEX.LT.303.OR.INDEX.GT.310) GOTO 35
       DO 36 J=1,15
       ITEMP(200+J)=BLANK
       ITEMP(250+J)=BLANK
       ITEMP(300+J)=BLANK
   36  CONTINUE
   35  CONTINUE
C
      IF(INDEX.LE.100.OR.INDEX.GE.600) GO TO 2040
      IF(INDEX.GT.100.AND.INDEX.LT.200) GO TO 100
      IF(INDEX.GT.200.AND.INDEX.LT.300) GO TO 200
      IF(INDEX.GT.300.AND.INDEX.LT.400) GO TO 300
      IF(INDEX.GT.400.AND.INDEX.LT.500) GO TO 400
      IF(INDEX.GT.500.AND.INDEX.LT.600) GO TO 500
C
C
  100 INDEX=INDEX-100
      GO TO(101,102,103,104,105,106,107,108,109,110,111,112,113,114,     330
     * 115,116,117,118,119,120),INDEX
  200 INDEX=INDEX-200
      GO TO(201,202,203,204,205,206,207,208,209,210,211,212,213,214,     333
```

2-20

```
CARD NO    ****                          CONTENTS                                  ****

               * 215,216,217,218,219,220,221,222,223,224,225,226,227,228,229,      334
               * 230,231,232,233,234,235,236,237,238,239,240,241,242,243),INDEX    335
           300 INDEX=INDEX-300
                GO TO(301,302,303,304,305,306,307,308,309,310,311,312,313),INDEX  337
           400 INDEX=INDEX-400
                GO TO(401,402,403,404,405,406,407,408,409,410,411,412,413,414,     339
               * 415,416,417,418,419,420,421,422,423,424,425,426,427,428,429,      340
               * 430,431,432,433,434,435,436,437,438,439,440,441,442,443,444,      341
               * 445,446,447,448,449,450,451,452,453,454,455,456,457,458,459,
               * 460,461,462,463,464,465,466,467,468,469,470),INDEX
           500 INDEX=INDEX-500
                GO TO(501,502,503,504,505,506,507,508,509,510,511,512,513,
               *  514,515),INDEX
          C
          1001 READ(5,NL101)
                IF(IADD1.LE.129.AND.IADD1.GE.1) GO TO 1101
                IADD1=1
          1101 CONTINUE
                CALL SECLRX(N00001,N00013,N00129,NRAND,XRNDM)
          1102 CONTINUE
                SIG2SQ=-2.0*SIGMA*SIGMA
                UL=UMEAN-0.5*UEXT
                XUEXT=UEXT/3.43597384E10
                IF(JRND.LE.0) JRND=1
                CALL SECLRX(N00021,N00001,N00012,ITEMP,XRNDM)
                WRITE(6,NL101)
                GO TO 1000
          C
          1002 READ(5,NL102)
```

1) $\boxed{a = b}\ \overline{Fa}$ produces $\boxed{a = b}\ \boxed{\overline{Fa}}\ \overline{Fb}$

2) $\boxed{\overline{Fa}}\ (a = b)$ produces $\boxed{\overline{Fa}}\ \boxed{a = b}\ Fb$

Though reduction is defined as a post processing operation, it is, in reality, more of a resolution operation and, therefore, should be executed within the resolution procedure or prior to other post processing operations.

(6) Ordering

Literals to the right of the right-most framed literal are free to be reordered. Reordering may not occur across framed literals without special care.

2.4 FACT DETERMINATION

OL-deduction, as has been stated, is a refutation procedure. A questionable assertion, that is, a query, is presented in negated form to the OL-deduction mechanism. Deductions are then generated and are sought to be refuted. A refutable deduction implies the existance of a collection of rules and facts (possibly a null collection) that refutes the negated query, and consequently, satisfies the assertion of the query. It is the collection of facts that is the answer to the positive query and that is provided by the inference system.

A resolved deduction is a clause consisting of only resolution literals; that is, it is a deduction clause that has been fully resolved.

Each resolution (framed) literal in a resolved deduction clause is a literal about which the information system can supply facts or is an acceptable inferred literal. An inferred literal will often be a simple binary proposition, signifying a yes or no; true or false; 0 or 1 value. In the more general case, where the framed literal is a complex predicate, the literal will tend to be a fact literal for which the information system will supply all facts. For example, given $\boxed{L(X,5)}$ where L(X,5) means "X" is a worker in factory "5", then the information system will search out all workers in factory number 5 and supply these as facts satisfying the literal. We call this procedure "fact determination" When fact determination is applied to all of the fact literals of a completely resolved deduction, we then determine a largest subset of these facts that can satisfy the deduction as an entirety.

```
 CARD NO       ****                          CONTENTS                                    ****

   464           208 READ(5,NL208)
   465               WRITE(6,NL208)
   466               GO TO 1206
   467           209 READ(5,NL209)
   468               WRITE(6,NL209)
   469          1206 IF( NHIST .LE. 8000 ) GO TO 1000
   470               WRITE(6,1210)
   471          1210 FORMAT( ' THE SPECIFIED NUMBER OF POINTS IN HISTOGRAM EXCEEEDS
   472              *8000  .....  NHIST SET TO 8000 ' )
   473               NHIST = 8000
   474               GO TO 1000
   475         C
   476         C
   477           210 CONTINUE
   478               READ(5,NL210)
   479               WRITE(6,NL210)
   480               GO TO 1000
   481           211 READ(5,NL211)
   482               WRITE(6,NL211)
   483               GO TO 1000
   484           212 CONTINUE
   485               READ(5,NL212)
   486               WRITE(6,NL212)
   487               GO TO 1000
   488           213 READ(5,NL213)
   489               WRITE(6,NL213)
   490               GO TO 1000
   491         C
   492           214 READ(5,NL214)
```

```
            WRITE(6,NL214)
            GO TO 1000
        215 READ(5,NL215)
            WRITE(6,NL215)
            GO TO 1000
        216 READ(5,NL216)
            WRITE(6,NL216)
            GO TO 1217
        217 READ(5,NL217)
            WRITE(6,NL217)
       1217  IF(ADCFS.EQ.0.0) GOTO 3217
             N=IFIX(FS/ADCFS+0.49)
             ERFS=ABS((ADCFS-(FS/FLOAT(N)))/FS)
             IF(ERFS.LT.1.0E-04) GOTO 1000
             ADCFS=FS/FLOAT(N)
             WRITE(6,2217) ADCFS
       2217  FORMAT(' THE SPECIFIED VALUE OF  ( FS/ADCFS ) NOT',
           * ' AN INTEGER ...ADCFS SET EQUAL TO', E13.5)
             GOTO 1000
       3217  ADCFS=FS
            GO TO 1000
        218 READ(5,NL218)
            WRITE(6,NL218)
            GO TO 1000
        219 READ(5,NL219)
            WRITE(6,NL219)
            GO TO 1000
      C
        220 READ(5,NL220)
```

```
0?/11/75        INPUT LISTING                          AUTOFLOW CHART SET - FWD/SCL  RADSIM
                                                                                      2-24
 CARD NO        ****                              CONTENTS                            ****

 [illegible]              WRITE(6,NL220)
 [illegible]              GO TO 1000
 [illegible]          221 READ(5,NL221)
 [illegible]              WRITE(6,NL221)
 [illegible]              GO TO 1000
 [illegible]          222 READ(5,NL222)
 [illegible]              WRITE(6,NL222)
 [illegible]              GO TO 1000
 [illegible]          223 READ(5,NL223)
 [illegible]              WRITE(6,NL223)
 [illegible]              GO TO 1000
 [illegible]          224 READ(5,NL224)
 [illegible]              WRITE(6,NL224)
 [illegible]              GO TO 1000
 [illegible]          225 READ(5,NL225)
 [illegible]              WRITE(6,NL225)
 [illegible]              GO TO 1000
 [illegible]          226 READ(5,NL226)
 [illegible]              WRITE(6,NL226)
 [illegible]              GO TO 1000
 [illegible]          227 READ(5,NL227)
 [illegible]              WRITE(6,NL227)
 [illegible]              GO TO 1000
 [illegible]          229 READ(5,NL229)
 [illegible]              WRITE(6,NL229)
 [illegible]              GO TO 1000
 [illegible]          230 READ(5,NL230)
 [illegible]              WRITE(6,NL230)
                          GO TO 1000
```

```
      231 READ(5,NL231)
          WRITE(6,NL231)
          GO TO 1000
      232 READ(5,NL232)
          WRITE(6,NL232)
          GO TO 1000
     233   READ(5,NL233)
           WRITE(6,NL233)
           GOTO 1000
     234   READ(5,NL234)
           WRITE(6,NL234)
           GOTO 1000
     235   READ(5,NL235)
           WRITE(6,NL235)
           GOTO 1000
     236   READ(5,NL236)
           WRITE(6,NL236)
           GOTO 1000
     237   READ(5,NL237)
           WRITE(6,NL237)
           GOTO 1000
C
      301 CONTINUE
          ICFOR=1
          ICINV=1
          READ(5,NL301)
           IF(FS.NE.0.0) T1=1.0/FS
           IF(T1.NE.0.0.AND.FS.EQ.0.0) FS=1.0/T1
          IF(FREQ.NE.0.0) LAMBDA= 0.299792E/FREQ
```

```
 CARD NO.        ****                           CONTENTS                              ****

   580                   IF(FI.LE.0.0) FI=FS/FLOAT(2**N2)
   581                  IF(SIMBW.LE.0.0.OR.SIMBW.GT.FS) SIMBW=FS
   582            1801   IF(RNGCEL.LE.0.0.AND.SIMBW.NE.0.0) RNGCEL=1.0/SIMBW
   583                   IF(RNGCEL.LT.TI) RNGCEL=TI
   584                  AZANG=ANTAZO+WSCAN*TIME
   585                  ELANG=ANTELO
   586                   IF(FI.LE.0.0) GO TO 2301
   587                  NF=IFIX(SIMBW/FI)
   588                  IF(NF.LE.8192)GO TO 2301
   589                  WRITE(6,3301)
   590            3301 FORMAT(' THE REQUIRED NUMBER OF FREQ. DOMAIN SAMPLES EXCEEDS'/
   591                 *,' 8192.....THIS JOB WILL BE TERMINATED')
   592                  GO TO 1797
   593            2301 WRITE(6,NL301)
   594                    GOTO 1000
   595            5301   WRITE(6,11) NREPET
   596                    IFC=FLD(6,6,IEXEC(1))
   597                    READ(11'IFC) ITEMP
   598            6301   IREPT=IREPT+1
   599                    IF(IREPT.GT.NREPET) GOTO 1801
   600                    IBLOCK=IBLOCK+1
   601                    IFC=IBLOCK
   602                    IPRIPT=IPRIPT+1
   603                    IF(IPRIPT.GT.NPRIS) IPRIPT=1
   604                    TIME=TIME+PRI(IPRIPT)
   605                    AZANG=ANTAZO+WSCAN*TIME
   606                    WRITE(11'IFC) ITEMP
   607                    FLD(6,6,IEXEC(1))=IFC
   608                    GOTO 1800
```

```
  11    FORMAT(' THIS CONFIGURATION WILL BE REPEATED',I3,
     *  'TIMES')
  302 READ(5,NL302)
      IF(KNGCEL.LT.II) KNGCEL=II
      CALL CLINI($1302)
      WRITE(6,NL302)
      GO TO 1000
  303 READ(5,NL303)
      WRITE(6,NL303)
      GO TO 1000
  304 READ(5,NL304)
      WRITE(6,NL304)
      GO TO 1000
  305 READ(5,NL305)
      WRITE(6,NL305)
      GO TO 1000
  306 READ(5,NL306)
      WRITE(6,NL306)
      GO TO 1000
  307 READ(5,NL307)
      WRITE(6,NL307)
      GO TO 1000
  308 READ(5,NL308)
      WRITE(6,NL308)
      GO TO 1000
  309 READ(5,NL309)
      WRITE(6,NL309)
      GO TO 1000
  310 READ(5,NL310)
```

```
                                                                        AUTOFLOW CHART SET - FWL/SCL  RADSIM
06/11/75          INPUT LISTING                                                                       2-28
                                                                                                       ****
                                                                CONTENTS
  CARD NO          ****

    638                    WRITE(6,NL310)
    639                    GO TO 1000
    640           C
    641              401 READ(5,NL401)
    642                    WRITE(6,NL401)
    643                    GO TO 1000
    644              402 READ(5,NL402)
    645                    WRITE(6,NL402)
    646                    GO TO 1000
    647           C
    648              403 READ(5,NL403)
    649                    WRITE(6,NL403)
    650                    GO TO 1000
    651              404 READ(5,NL404)
    652                    WRITE(6,NL404)
    653                    GO TO 1000
    654              405 READ(5,NL405)
    655                    WRITE(6,NL405)
    656                    GO TO 1000
    657              406 READ(5,NL406)
    658                    WRITE(6,NL406)
    659                    GO TO 1000
    660           C
    661              407 READ(5,NL407)
    662                    WRITE(6,NL407)
    663                    GO TO 1000
    664              408 READ(5,NL408)
    665                    WRITE(6,NL408)
    666                    GO TO 1000
  409 READ(5,NL409)
      WRITE(6,NL409)
      GO TO 1000
  410 READ(5,NL410)
      WRITE(6,NL410)
      GO TO 1000
  413 READ(5,NL413)
      WRITE(6,NL413)
      GO TO 1000
C
  420 READ(5,NL420)
      WRITE(6,NL420)
 1420  IF(RISTIM.LT.TI) RISTIM=TI
       IF(FALTIM.LT.TI) FALTIM=TI
       IF(SWTIM.LT.TI) SWTIM=TI
      GO TO 1000
  421 READ(5,NL421)
      WRITE(6,NL421)
      GO TO 1420
C
  422 READ(5,NL422)
      WRITE(6,NL422)
      GO TO 1000
  423 READ(5,NL423)
      WRITE(6,NL423)
      GO TO 1000
C
  425 READ(5,NL425)
      WRITE(6,NL425)
```

```
CARD NO.   ****                         CONTENTS                         ****

                IF(NF.GT.4000) GOTO 2426
                GO TO 1000
          426   READ(5,NL426)
                WRITE(6,NL426)
          1426  IF(MODISK.EQ.3) GOTO 1000
                IF(NF.LE.4000) GOTO 1000
          2426  WRITE(6,3426)
          3426  FORMAT(' THE NUMBER OF FREQ. DOMAIN SAMPLES EXCEEDS 4000...',
               * ' THIS JOB WILL BE TERMINATED')
                GO TO 1797
          427   READ(5,NL427)
                WRITE(6,NL427)
                GO TO 1820
          430   READ(5,NL430)
                WRITE(6,NL430)
                GO TO 1830
          431   READ(5,NL431)
                WRITE(6,NL431)
                GO TO 1830
          432   READ(5,NL432)
                WRITE(6,NL432)
                GO TO 1830
          433   READ(5,NL433)
                WRITE(6,NL433)
          1830  CONTINUE
                IF(IFFOD.EQ.0) IFFOD=1
                IF(IFF1D.EQ.0) IFF1D=1
                IF(IFB1D.EQ.0) IFB1D=1
                IF(IFB2D.EQ.0) IFB2D=1
```

```
      GO TO 1000
  434 READ(5,NL434)
      WRITE(6,NL434)
      GO TO 1000
  435 READ(5,NL435)
      WRITE(6,NL435)
      GO TO 1000
  436 READ(5,NL436)
      WRITE(6,NL436)
      GO TO 1000
  437 READ(5,NL437)
      WRITE(6,NL437)
      GO TO 1000
  440 READ(5,NL440)
      WRITE(6,NL440)
      GO TO 1000
  441 READ(5,NL441)
      WRITE(6,NL441)
      GO TO 1000
  451 READ(5,NL451)
      WRITE(6,NL451)
      GO TO 1000
  452 READ(5,NL452)
      WRITE(6,NL452)
      GO TO 1000
453      READ(5,NL453)
         WRITE(6,NL453)
         GOTO 1000
454      READ(5,NL454)
```

```
CARD NO         ****                                CONTENTS

                       WRITE(6,NL454)
                       GO TO 1000
                455    READ(5,NL455)
                       WRITE(6,NL455)
                       GO TO 1000

                456    READ(5,NL456)
                       WRITE(6,NL456)
                       GO TO 1000
                457    READ(5,NL457)
                       WRITE(6,NL457)
                       GO TO 1000
                458    READ(5,NL458)
                       WRITE(6,NL458)
                       GO TO 1000
                459    READ(5,NL459)
                       WRITE(6,NL459)
                       IF(IAVG.LT.II*2.0) IAVG II*2.0
                       GO TO 1000
                460    READ(5,NL460)
                       WRITE(6,NL460)
                       GO TO 1455
                461    READ(5,NL461)
                       WRITE(6,NL461)
                       GO TO 1000
                462    READ(5,NL462)
                       WRITE(6,NL462)
                       GO TO 1000
                463    READ(5,NL463)
```

```
      WRITE(6,NL463)
      GOTO 1000
C
  501 READ(5,NL501)
      WRITE(6,NL501)
      GO TO 1000
  502 READ(5,NL502)
      WRITE(6,NL502)
      GO TO 1000
  504 CONTINUE
      READ(5,NL504)
      WRITE(6,NL504)
      GO TO 1000
  505 READ(5,NL505)
      WRITE(6,NL505)
       IF(NF.GT.4000) GOTO 2426
      GO TO 1000
C
  506   READ(5,NL506)
        WRITE(6,NL506)
        CALL PHENC($1500)
        MODULE=420
        GOTO 1420
  507   READ(5,NL507)
        WRITE(6,NL507)
        CALL PHENC($1500)
        MODULE=421
        GOTO 1420
  508   MODEL=SIMFO
```

```
CARD NO        ****                               CONTENTS

                    READ(5,NL508)
                    WRITE(6,NL508)
                    IWRITE=1
                    GO TO 1000
                    [illegible] IMFO
                    READ(5,NL509)
                    WRITE(6,NL509)
                    IWRITE=1
                    GO TO 1000
                    READ(5,NL510)
                    WRITE(6,NL510)
                      GO TO 1000
                    READ(5,NL511)
                    WRITE(6,NL511)
                    GO TO 1000
                    READ(5,NL512)
                    WRITE(6,NL512)
                      GO TO 1000
```

```
  204 CONTINUE
  205 CONTINUE
  206 CONTINUE
  207 CONTINUE
  228 CONTINUE
 230    CONTINUE
C
  312 CONTINUE
  313 CONTINUE
C
  414 CONTINUE
  415 CONTINUE
  416 CONTINUE
  417 CONTINUE
 418    CONTINUE
  419 CONTINUE
  438 CONTINUE
  439 CONTINUE
  442 CONTINUE
  443 CONTINUE
  445 CONTINUE
  446 CONTINUE
  447 CONTINUE
  448 CONTINUE
  449 CONTINUE
  450 CONTINUE
C
  503 CONTINUE
C
```

```
[illegible]        WRITE(6,907) JSTEP,MODULE
[illegible]        FORMAT(1H0, ' NOTHING LOADED FOR STEP NUMBER',I4, '.....THE MODULE665
[illegible]        [illegible] I,A,I4)
[illegible]        [illegible]
[illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
[illegible]        [illegible]
```

```
      513   CONTINUE
      514   CONTINUE
      515   CONTINUE
      2040 WRITE(6,51) JSTEP,MODULE
        51 FORMAT( ' THE MODULE NUMBER FOR THE',I3,'TH STEP IS INVALID...THE 699
          * MODULE NUMBER IS ',I4)
           JSTEP=JSTEP-1                                                      7042
           IBLOCK=IBLOCK-1                                                    7043
           GO TO 2000
     C        ** LOAD DATA BLOCK AND EXEC WORD **
      1000  WRITE (11'IFC) ITEMP
      1001 ITEXEC= MODULE
           ITEXEC= IPACK(12,ICW,ITEXEC)
           ITEXEC= IPACK(24,IFC,ITEXEC)
           IEXEC(JSTEP)= IPACK(30,ICFG,ITEXEC)
           GO TO 2000
     C                          ** ENDPAS **
      1500  IBLOCK=IBLOCK+1
            IEXEC(500)=IBLOCK
            WRITE (11'IBEXEC) IEXEC
            WRITE (6,61) IBEXEC,IBLOCK
      61    FORMAT(' IBEXEC = ',I10,' IBLOCK = ',I10)
            IBEXEC=IBLOCK
           ISIM=ISIM+1                                                    730
            IF(NREPET.EQ.0) GOTO 1501
            IF(IREPT.EQ.0) GOTO 5301
            GOTO 6301
      1501  IFLAGG=1
           GO TO 2000
```

```
 CARD NO        ****                              CONTENTS                                    ****

   928           1530 WRITE(6,1535)
   929           1535 FORMAT( ' MORE THAN 250 JOB STEPS PER SIMULATION PASS ATTEMPTED...734
   930               * THE SIMULATION ACTIVITY IS DELETED')
   931                GO TO 1610
   932           C
   933           1605 INDEX = MODULE
   934                ICW = 1
   935                 IFC=IBLOCK
   936                 IF(IFCFLG.EQ.0) IFC=IHEXEC-1
   937                JSTEP = JSTEP+1
   938                IF( JSTEP .GT. 250 ) GO TO 1530
   939                GO TO 2050
   940           C
   941           1606 ICW=1
   942                GO TO 1611
   943           1610 ICW=2
   944                 IFCFLG=1
   945                IBLOCK = IBLOCK+1
   946                IF(IBLOCK.GT.40) GO TO 1799
   947           1611 INDEX = MODULE
   948                JSTEP=JSTEP+1
   949                IF( JSTEP.GT.250 ) GO TO 1530
   950                 IFC=IBLOCK
   951                WRITE(6,1612) INDEX                                                    753
   952           1612 FORMAT(1H0,' THE NEXT BLOCK OF DATA TO BE LOADED IS FOR MODULE NUM754
   953               *BER', I4 //  )                                                        755
   954                GO TO 51                                                               756
   955           C
   956           2050 CONTINUE                                                                758
```

```
      ICW=4
      GO TO 1001
 1620 CONTINUE
      ICW=5
      GO TO 1001
 1625 CONTINUE
      ICW=6
      GO TO 1001
 1630 CONTINUE
      ICW=7
      GO TO 1001
 1635 CONTINUE
      ICW=8
      GO TO 1001
 1640 CONTINUE
      ICW=9
      GO TO 1001
 1645 CONTINUE
      ICW=10
      GO TO 1001
 1650 CONTINUE
      ICW=11
      GO TO 1001
 1655 CONTINUE
      ICW=3
      GO TO 1001
 1660 CONTINUE
      ICW=12
      GO TO 1001
 CARD NO        ****                                CONTENTS                  2-40****

   986          1665 CONTINUE
   987               ICW=13
   988               GO TO 1001
   989          1670 CONTINUE
   990               ICW=14
   991               GO TO 1001
   992          1675 CONTINUE
   993               ICW=15
   994               GO TO 1001
   995          1680 CONTINUE
   996               ICW=16
   997               GO TO 1001
   998          1685 CONTINUE
   999               ICW=17
  1000               GO TO 1001
  1001          1690 CONTINUE
  1002               ICW=18
  1003               GO TO 1001
  1004          1695 CONTINUE
  1005               ICW=19
  1006               GO TO 1001
  1007          1730 CONTINUE
  1008               ICW=20
  1009               GO TO 1001
  1010          1735 CONTINUE
  1011               ICW=21
  1012               GO TO 1001
  1013          1740 CONTINUE
  1014               ICW=22
```

```
      GO TO 1001
 1745 CONTINUE
      ICW=23
      GO TO 1001
C                          ** ENDCFG **
 1700 CONTINUE
      ISIM=ISIM-1
      WRITE(6,1701) ICFG,ISIM
 1701 FORMAT(1H0,'THE DATA LOAD FOR CONFIGURATION NUMBER',I3,' HAS BEEN
     *COMPLETED.....',I3,' SIMULATION PASSES WERE LOADED')
       ICFG=ICFG+1
      GO TO 5
C                          ** MODIFY **
 1710 CONTINUE
      JSTEP = 111
      INDEX=MODULE
       ICW=FLD(12,12,IEXEC(JSTEP))
       IFC=FLD(6,6,IEXEC(JSTEP))
       IF(ICARD.EQ.SKIP) WRITE(6,1751) JSTEP
 1751  FORMAT('  STEP NUMBER',I3,' WILL BE BYPASSED FOR REMAINDER',
     *  ' OF THIS CONFIGURATION')
      IF(ICARD.EQ.MODIFY) WRITE(6,1712) ISIM,IFC   ,JSTEP,MODULE
 1712 FORMAT(1H ,'MOD FOR PASS NUMBER',I3,' DATA BLOCK',I3,' FOR J
     *OB STEP',I4,' CALLING MODULE',I4,' IS TO BE MODIFIED' )
C
      IF(  IFC  .NE.0 ) GO TO 1720
      WRITE(6,1714)
 1714 FORMAT( ' IBLOCK = 0  MODIFICATION NOT PERFORMED.' )
       GO TO 2000
 CARD NO      ****                              CONTENTS                        ****

              1720  READ (11'IFC) ITEMP
                    IBLOCK=IBLOCK+1
                    IFC=IBLOCK
                    IF(ICARD.EQ.MODIFY) GOTO 35
                    MODULE=0
                    GOTO 1001
              C                                                                851
               1302 WRITE(6,2302) ICFG                                          852
               2302 FORMAT(' NON-STANDARD RETURN FROM CLINT.....THE CONFIGURATIONS PRE853
                   *CEEDING',I5,' WILL BE EXECUTED')                             854
                    GO TO 1797                                                  855
               1506 WRITE(6,2506)                                               856
               2506 FORMAT(' NON-STANDARD RETURN FROM PHENC....PRECEEDING            8520
                   * CONFIGURATIONS WILL BE EXECUTED')                              8521
                    GO TO 1797
               1799 WRITE(6,1798) ICFG                                          859
               1798 FORMAT( ' THE NUMBER OF DATA BLOCKS TO BE LOADED EXCEEDS STORAGE A860
                   *VAILABLE.....THE CONFIGURATIONS PRECEEDING',I5,' WILL BE EXECUTED'861
                   *)                                                           862
               1797 CONTINUE
               1800 CONTINUE
                    ICFG=ICFG-1
                    WRITE(6,1601) ICFG                                          866
               1601 FORMAT(1H0,'THE DATA LOAD FOR',I3,' CONFIGURATIONS HAS BEEN COMPLE867
                   *TED.....SIMULATION ACTIVITY FOLLOWS')                        868
               1510  IEXEC(1)=0
                     WRITE (11'IBEXEC) IEXEC
              C                                                                873
                    CALL EXIT                                                   874
              C                                                                875
                    STOP                                                        876
                     END
```

SECTION 3

SIMULATION CONTROLLER

(MAIN - 2)

The simulation controller serves as the switchboard which connects the simulation modules together in the manner prescribed by the control cards read by the simulation data loader. Figure 3-1 is a block diagram of the Simulation Load Module. The control word blocks and System/Module Parameter Tables are stored in DSRN #11 which was initialized during the simulation data loader activity. The clutter scatterer parameters are stored in DSRN #2 which was initialized during the simulation data loader activity. In addition, provisions are available for reading and writing data on user defined disc and magnetic tape data sets.

There are two storage arrays allocated for storing simulated signal data. These arrays are designated as XT and YT. In addition, two auxiliary arrays are allocated for storing either signal data, processed data such as probability distributions, or other miscellaneous data. These arrays are designated as XA and XB. All four arrays are composed of 8192 elements.

The following is a description of the sequence of events that occur in performing a step of the simulation.

1. The operation to be performed in the Jth step is defined by the control word stored in location J of the control word block. This word is compared with the control word dictionary to determine (1) the operation to be performed, (2) the reference number of the module to be executed, and (3) the Data Set Reference Number.

2. If the operation to be performed is the execution of a module with no input data requirements, then control is transferred to the designated module. If the operation to be performed is execution of a module requiring input data, then the System/Module Parameter table is loaded from DSRN #11 and control is transferred to the designated module. For operations involving data transfers the contents of the designated array are either loaded from or transferred to the DSRN which was specified.

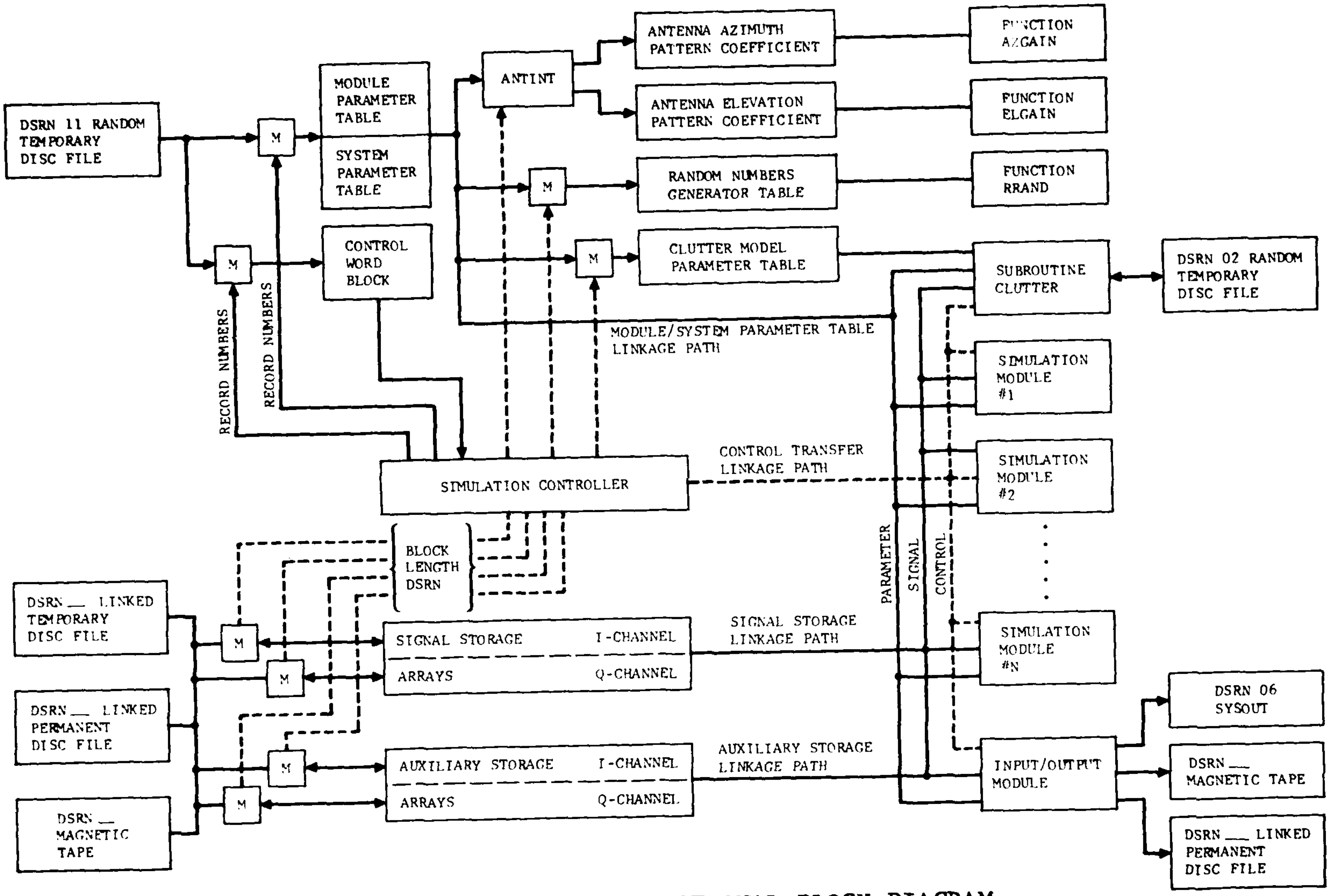


Figure 3-1 SIMULATION LOAD MODULE FUNCTIONAL BLOCK DIAGRAM

3. Once the operation for the Jth step is completed the job step counter is incremented by 1 and the process is repeated.

When a blank (all zeros) control word is encountered, the simulation pass is ended. The job step counter is reset to 1 and a new pass through the simulation is begun with a different set of parameters. A block of data containing all zeros is used to designate the end of a simulation configuration. When this is encountered the simulation controller will move the next block of data into the control word buffer. This new block of control words defines a new simulation configuration. When a new control word block containing all zeros is encountered, the simulation job is terminated.

Flow Chart: Page 9-34

Cross Reference Table: Page 9-210

3-4

```
C         * MAIN 2 *
         COMMON IT,TORIG,DELT,TDUM,XT(8193),ITY,TORIGY,DELTY,TDUMY,
        * YT(8193),IXA,XAORIG,XADEL,XADUM,XA(8193),IEXEC(500)
         COMMON/BLK1/IDATA(500)
         COMMON/BLK330/ICLCON(30)
         COMMON/AZPAT/ COEFAZ(304)
         COMMON/ELPAT/ COEFEL(304)
         COMMON/BL0002/ IXB,XBORIG,XBDEL,XBDUM,XB(8193)
         COMMON/BLKEND/ KNDDAT(141)                                80
          COMMON/SYS/ MODULE,ICW,IFC
C
         DIMENSION XTT(516),YTT(516),XAT(516),XBT(516)             100
C
         DIMENSION XD1(4010),XD2(4010),XB1(4010),XB2(4010)
         DIMENSION IXT(8192),IYT(8192)
         DIMENSION ANTAZ(150),ANTEL(150)
         DIMENSION XTM(4010),YTM(4010)
C
         DATA AXT/'  XT  '/, AYT/'  YT  '/, AXA/'  XA  '/, AXB/'  XB  '/       180
         DATA N193,N194,N195,N196/-3,-2,-1,0/
C
          CHARACTER LABL*16
          EQUIVALENCE ( IBEXEC, IEXEC(500))
          EQUIVALENCE (IDATA(316),LABL)
         EQUIVALENCE (XT(1),IXT(1)),(YT(1),IYT(1))
         EQUIVALENCE  ( XTT(1),  IT      ), ( YTT(1),  ITY     ),
 CARD NO      ****                          CONTENTS                               ****

                 *              ( XAT(1),  IXA     ), ( XBT(1),  IXB     )
                  EQUIVALENCE (XD1(1),XA(1)),(XD2(1),XA(4011))
                   EQUIVALENCE (XB1(1),XB(1)),(XB2(1),XB(4011))
                  EQUIVALENCE (XTM(1),XT(4011)),(YTM(1),YT(4011))
                  EQUIVALENCE  (IDATA(150), AZBST ) , (IDATA(151),NPTAZ),
                 *              (IDATA(152), ELEST ) , (IDATA(153),NPTEL),
                 *              (IDATA(201),ANTAZ(1)), (IDATA(351),ANTEL(1))
                  EQUIVALENCE (IDATA(160), MODEDF )
                  EQUIVALENCE   (IDATA(  1), N2        ),(IDATA(  2), FS        ),
                 *               (IDATA(  3), REFD      ),(IDATA(  4), SIMBW     ),
                 *               (IDATA(  5), ISDUMP    ),(IDATA(  6), ICINV     ),
                 *               (IDATA(  7), ICPLR     ),(IDATA(  8), SIMFD     ),
                 *               (IDATA(  9), NORMPT    ),(IDATA( 10), WSCAN     ),
                 *               (IDATA( 11), FI        ),(IDATA( 12), TI        ),
                 *               (IDATA( 13), LAMBDA    ),(IDATA( 14), RNGDEL    ),
                 *               (IDATA( 15), PRF       ),(IDATA( 16), TIME      ),
                 *               (IDATA( 17), AZANG     ),(IDATA( 18), ANTAZO    ),
                 *               (IDATA( 19), ELANG     ),(IDATA( 20), ANTELO    )
              C
                  DATA N00000,N00001,N00012,N00021,N00013,N00129,N00201
                 * /0,1,12,21,13,129,201/
              C
                   CALL RANSIZ(11,500)
                   IDEXEC=1
                  ICLCON(20)=0
                  CALL PACPT(06,1,1,0)                                                     480
                  ISDUMP=1
             290   READ(11'IDEXEC) IEXEC
                   WRITE(6,290) IDEXEC
```

```
       593    FORMAT( ' IBEXEC= ',I10)
              WRITE(6,595) ( J, IEXEC(J), J=1,250 )
       595 FORMAT(1H ,5(2X, 'IEXEC(', I3, ')= ',O12))                        550
       594    IPASS=1
              ICFG=IFLD(0,6,IEXEC(1))
               IF(ICFG.EQ.0.AND.IBEXEC.EQ.1) GO TO 2060
               IF(ICFG.EQ.0) GO TO 2050
              GO TO 999
        900 CONTINUE
              WRITE(6,71) IPASS,ICFG
          71 FORMAT(1H0,' SIMULATION PASS NUMBER',I3,' OF CONFIGURATION NUMBER'
            *,I3,' HAS BEEN COMPLETED')
              IPASS=IPASS+1
               READ(11'IBEXEC) IEXEC
               WRITE(6,596) IBEXEC
               IF(ISDUMP.EQ.1.OR.ISDUMP.EQ.2) WRITE(6,595) (J,IEXEC(J),J=1,250)
               IF(ICFG.NE.FLD(0,6,IEXEC(1))) GO TO 594
        999 CONTINUE
              WRITE(6,70) IPASS,ICFG
          70 FORMAT(1H1,' SIMULATION PASS NUMBER',I3,' OF CONFIGURATION NUMBER'
            *,I3,' IS BEGINNING')
              JSTEP=0
      C
       1000 JSTEP=JSTEP+1
              MODULE=FLD (24,12,IEXEC(JSTEP))                                730
              ICW=FLD (12,12,IEXEC(JSTEP))                                   740
              IFC=FLD (0,6,IEXEC(JSTEP))
              IF(ICW.EQ.0) GO TO 900
              ITEMP=MODULE
```

3-7

```
 CARD NO      ****                           CONTENTS                              ****

   1160              GO TO (951,952,953,954,955,956,957,958,959,960,961,962,963,964,
   1161             * 965,966,967,968,969,970,971,972,973),ICW
   1162       951     CONTINUE
   1163       952     TIMEOL=TIME
   1164               AZANGO=AZANG
   1165               ELANGO=ELANG
   1166               READ(11'IFC) IDATA
   1167               IF(MODULE.EQ.301) GOTO 988
   1168               TIME=TIMEOL
   1169               AZANG=AZANGO
   1170               ELANG=ELANGO
   1171       988     CONTINUE
   1172              WRITE(6,2041)
   1173      2041 FORMAT (1H1)
   1174              WRITE(6,2042) MODULE,ICW,IFC,ICFG
   1175      2042 FORMAT(' MODULE=',I10,'  ICW=',I10,'  IFC=',I10,'  ICFG=',I10)
   1176               IF(ISDUMP.GE.2) WRITE(6,2040) (J,IDATA(J), J=1,500)
   1177      2040 FORMAT(1H ,5(2X, 'IDATA(', I3, ')= ',O12))                        900
   1178              WRITE(6,10) ITEMP
   1179        10 FORMAT(' THE NEXT SCHEDULED OPERATION HAS AN EXEC NUMBER OF ',I4)
   1180      C
   1181               IF(ITEMP.LT.303.OR.ITEMP.GT.310) GOTO 998
   1182               ENCODE(LABEL,37) IPASS,ICFG
   1183        37     FORMAT(' PASS=',I2,'    CFG=',I2)
   1184       998     CONTINUE
   1185              IF( ITEMP .LE. 100 .OR. ITEMP .GE. 600 ) GO TO 2100
   1186              IF(ITEMP.GT.100.AND.ITEMP.LT.200) GO TO 100
   1187              IF(ITEMP.GT.200.AND.ITEMP.LT.300) GO TO 200
   1188              IF(ITEMP.GT.300.AND.ITEMP.LT.400) GO TO 300
```

```
      IF(ITEMP.GT.400.AND.ITEMP.LT.500) GO TO 400
      IF(ITEMP.GT.500.AND.ITEMP.LT.600) GO TO 500
C
C
  100 ITEMP=ITEMP-100
      GO TO(101,102,103,104,105,106,107,108,109,110,111,112,113,114,
     * 115,116,117,118,119,120),ITEMP
  200 ITEMP=ITEMP-200
      GO TO(201,202,203,204,205,206,207,208,209,210,211,212,213,214,
     * 215,216,217,218,219,220,221,222,223,224,225,226,227,228,229,
     * 230,231,232,233,234,235,236,237,238,239,240,241,242,243),ITEMP
  300 ITEMP=ITEMP-300
      GO TO(301,302,303,304,305,306,307,308,309,310,311,312,313),ITEMP
  400 ITEMP=ITEMP-400
      GO TO (401,402,403,404,405,406,407,408,409,410,411,412,413,414,
     * 415,416,417,418,419,420,421,422,423,424,425,426,427,428,429,
     * 430,431,432,433,434,435,436,437,438,439,440,441,442,443,444,
     * 445,446,447,448,449,450,451,452,453,454,455,456,457,458,459,
     * 460,461,462,463,464,465,466,467,468,469,470),ITEMP
  500 ITEMP=ITEMP-500
      GO TO(501,502,503,504,505,506,507,508,509,510,511,512),ITEMP
C
C
  101 CONTINUE
      CALL SDBLKX(N00201,N00013,N00129,IDATA,RNDDAT)                   1240
  102 CALL SDBLKX(N00021,N00001,N00012,IDATA,RNDDAT)                   1250
      WRITE(6,1101) (J,RNDDAT(J), J=1,141)
 1101 FORMAT(1H , 4( 2X, 'RNDDAT(',I3, ') = ', O12))                   1270
      GO TO 1000
```

```
CARD NO      ****                               CONTENTS                                ****

              103 CALL RTOPDB(XT,YT,XA,XB)
                  GO TO 1000
              104 CALL XYTODE(XT,YT,XA)
                  GO TO 1000
              105 CALL XYTOM(XT,YT,XA)
                  GO TO 1000
              106 CALL XYTOM2(XT,YT,XA)
                  GO TO 1000
              108 CALL XYTODB(XT,YT,XT)
                  GO TO 1000
              110 CALL RTOPM(XT,YT,XA,XB)
                  GO TO 1000
              111 CALL RTOPM2(XT,YT,XA,XB)
                  GO TO 1000
              113 CALL PLTFMT(XT,YT,XA,$1070)                                         1430
                  GO TO 1000
            C
              114 CALL DBLKX(N193,E197,XT,XA)
                  CALL DBLKX(N193,E197,YT,XB)
                  GO TO 1000
            C
              115 CALL DBLKX(N193,E197,XA,XT)
                  CALL DBLKX(N193,E197,XB,YT)
                  GO TO 1000
            C
              116   CALL ERGYKE(XT)
                    GOTO 1000
              117   CALL EFCYKE(YT)
                    GOTO 1000
```

```
  118   CALL ENGYCP(XT,YT)
        GOTO 1000
C
   201 CALL DFT(XT,YT)
       GO TO 1000
   202 CONTINUE
       CALL ZFFT(XT,YT)
       GO TO 1000
   203 CONTINUE
       CALL ZIFFT(XT,YT)
       GO TO 1000
   204 CALL CONV(XT,YT,XA,XB,$1130)
       GO TO 1000
   205 CALL CONVMP(XT,YT,XA,XB,$1130)
       GO TO 1000
   206 CALL DIVA(XT,YT,XA,XB,$1130)
       GO TO 1000
   207 CALL ADDA(XT,XA,$1130)
       GO TO 1000
   208 CALL COMDIS(XT,XA)
       GO TO 1000
   209 CALL COMDIS(YT,XA)
       GO TO 1000
   210 CALL OUTCOM(XB,XA)
       GO TO 1000
   211 CALL OUTCOM(XA,XA)
       GO TO 1000
   212 CALL PDF(XB,XA)
       GO TO 1000
  CARD NO          ****                         CONTENTS                         ****

   1276             213 CALL PDF(XA,XA)
   1277                 GO TO 1000
   1278             214 CALL RNDARY(XT)
   1279                 GO TO 1000
   1280             215 CALL RNDARY(YT)
   1281                 GO TO 1000
   1282             216 CALL ATOD(XT,IXT)
   1283                 GO TO 1000
   1284             217 CALL ATOD(YT,IYT)
   1285                 GO TO 1000
   1286             218 CALL DTOA(IXT,XT)
   1287                 GO TO 1000
   1288             219 CALL DTOA(IYT,YT)
   1289                 GO TO 1000
   1290             220 CALL DTOA(IYT,XA)
   1291                 GO TO 1000
   1292             221 CALL WRITFE(XT,YT,$1010)
   1293                 GO TO 1000
   1294             222 CALL WRITCP(XT,YT,$1010)
   1295                 GO TO 1000
   1296             223 CALL WRITMP(XT,YT,$1010)
   1297                 GO TO 1000
   1298             224 CALL SHIFT(XT,YT,XT,YT)
   1299                 GO TO 1000
   1300             225 CALL SHIFT(XT,YT,XA,XB)
   1301                 GO TO 1000
   1302             226 CALL SHIFTS(XT,YT,XA,XB)
   1303                 GO TO 1000
   1304             227 CONTINUE
```

```
      GO TO 972
 1227 CONTINUE
      GO TO 973
 2227 CALL SHIFTS(XT,YT,XA,XB)
      GO TO 1000
  228 CALL DTOA(IXT,XA)
      GO TO 1000
  229 CALL RSHIFT(XT,YT,XT,YT)
      GO TO 1000
  230 CALL RSHIFT(XT,YT,XA,XB)
      GO TO 1000
  231 CALL RSHFTS(XT,YT,XA,XB)
      GO TO 1000
  232 CONTINUE
      GO TO 972
 1232 GO TO 973
 2232 CALL RSHFTS(XT,YT,XA,XB)
      GO TO 1000
  233   CALL DFTRF(XT,YT)
        GOTO 1000
  234   CALL DFTFO(XT,YT)
        GOTO 1000
  235   CALL ADDRND(XT)
        GOTO 1000
  236   CALL ADDRND(YT)
        GOTO 1000
  237   CALL ADRNDC(XT,YT)
        GOTO 1000
  238   CALL CONV(XT,YT,XL1,XL2,$1130)
 CARD NO        ****                        CONTENTS                           ****

   1334                GOTO 1000
   1335          239   CALL ADDA(YT,XB,$1130)
   1336                GOTO 1000
   1337         C
   1338           301 CONTINUE
   1339               IF(ICFG.GT.1.AND.NPTAZ.EQ.0) GO TO 1302
   1340               CALL ANTINT(NPTAZ,AZBST,ANTAZ,COEFAZ)
   1341          1302 IF(ICFG.GT.1.AND.NPTEL.EQ.0) GO TO 1000
   1342               CALL ANTINT(NPTEL,ELBST,ANTEL,COEFEL)
   1343               GO TO 1000
   1344           302 CALL SDBLKX(46,1,13,IDATA,ICLCON)
   1345               CALL SDBLKX(120,14,9,IDATA,ICLCON)
   1346               GO TO 1000
   1347           303 CALL PTLIST(XT)
   1348               GO TO 1000
   1349           304 CALL PTLIST(YT)
   1350               GO TO 1000
   1351           305 CALL PTLIST(XA)
   1352               GO TO 1000
   1353           306 CALL PTLIST(XB)
   1354               GO TO 1000
   1355           307 CALL PLOTTR(XT)
   1356               GO TO 1000
   1357           308 CALL PLOTTR(YT)
   1358               GO TO 1000
   1359           309 CALL PLOTTR(XA)
   1360               GO TO 1000
   1361           310 CALL PLOTTR(XB)
   1362               GO TO 1000
```

```
  312 CALL SPCAVG(XA,$1090)                                          2500 3-14
      GO TO 1000
  313 CALL SCANNR(XA)                                                2520
      GO TO 1000
C
C
  401 CALL NONLIN(XT,$1050)
      GO TO 1000
  402 CALL NONLIN(YT,$1050)
      GO TO 1000
 403   CALL CEIGFL(XT,YT)
      GO TO 1000
 404   CALL CDFNCL(XT,YT)
      GO TO 1000
 405   CALL REIGFL(XT,YT)
       GO TO 1000
 406   CALL RDFNCL(XT,YT)
       GO TO 1000
  407 CALL FILT(XT,YT)
      GO TO 1000
 408   CALL FILT(XA,XF)
       GO TO 1000
  409 CALL INGTCR(XT,XT)
      GO TO 1000
  410 CALL INGTCR(YT,YT)
      GO TO 1000
  411 CALL ANTARY(XT,YT,$1060)                                       2700
      GO TO 1000
  414 CALL KWDCI(XT,XT)
 CARD NO        ****                           CONTENTS                            ****

   1392                GO TO 1000
   1393            415 CALL HWDET(YT,YT)
   1394                GO TO 1000
   1395            416 CALL FWDET(XT,XT)
   1396                GO TO 1000
   1397            417 CALL FWDET(YT,YT)
   1398                GO TO 1000
   1399            418 CALL SWDET(XT,XT)
   1400                GO TO 1000
   1401            419 CALL SWDET(YT,YT)
   1402                GO TO 1000
   1403            420 CALL FGENXY(XT,YT,$1030)
   1404                GO TO 1000
   1405            421 CALL FGENMF(XT,YT,$1030)                                        2800
   1406                GO TO 1000
   1407            422 CALL DIGTFL(IXT,IXT)
   1408                GO TO 1000
   1409            423 CALL DIGTFL(IYT,IYT)
   1410                GO TO 1000
   1411            425 CALL CGEN(XT,YT,XD1,XD2,$1040)
   1412                GO TO 1000
   1413            426 CALL TSARY(XT,YT,XD1,XD2,$1020)
   1414                GO TO 1000
   1415            427 CALL TSARY1(XT,YT,XD1,XD2,$1020)
   1416                GO TO 1000
   1417            450 GO TO(1430,2430),MODEOF
   1418           1430 CALL MTIFLT(XT,XT)
   1419                GO TO 1000
   1420           2430 CALL MTIIFT(IXT,IXT)
```

```
      GO TO 1000
  431 GO TO(1431,2431),MODEDF
 1431 CALL MTIFLT(YT,YT)
      GO TO 1000
 2431 CALL MTIIFT(IYT,IYT)
      GO TO 1000
  432 GO TO(1432,2432),MODEDF
 1432 CALL MTINCL(XT,XT)
      GO TO 1000
 2432 CALL MTIINC(IXT,IXT)
      GO TO 1000
  433 GO TO(1433,2433),MODEDF
 1433 CALL MTINCL(YT,YT)
      GO TO 1000
 2433 CALL MTIINC(IYT,IYT)
      GO TO 1000
  434 CALL SWPINT(XT,XT)
      GO TO 1000
  435 CALL SWPINT(YT,YT)
      GO TO 1000
  436 CALL NCSWPI(XT,XT)
      GO TO 1000
  437 CALL NCSWPI(YT,YT)
      GO TO 1000
  438 CALL HLIM(XT,XT)
      GO TO 1000
  439 CALL HLIM(YT,YT)
      GO TO 1000
  440 CALL DCFAR(IXT,XA)
 CARD NO.     ****                        CONTENTS                        ****

   1450             GO TO 1000
   1451         441 CALL DCFAR(IYT,XB)
   1452             GO TO 1000
   1453         442 CALL IHLIM(IXT,IXT)
   1454             GO TO 1000
   1455         443 CALL IHLIM(IYT,IYT)
   1456             GO TO 1000
   1457         445 CALL IHWDET(IXT,IXT)
   1458             GO TO 1000
   1459         446 CALL IHWDET(IYT,IYT)
   1460             GO TO 1000
   1461         447 CALL IFWDET(IXT,IXT)
   1462             GO TO 1000
   1463         448 CALL IFWDET(IYT,IYT)
   1464             GO TO 1000
   1465         449 CALL ISGDET(IXT,IXT)
   1466             GO TO 1000
   1467         450 CALL ISGDET(IYT,IYT)
   1468             GO TO 1000
   1469         451 CALL RECF(XT,YT,XA,XB)
   1470             GO TO 1000
   1471         452 CALL RECFTF(XT,YT)
   1472             GO TO 1000
   1473         453   CALL CGENSF(XT,YT)
   1474               GOTO 1000
   1475         454   CALL FAFRM(XT,YT)
   1476               GOTO 1000
   1477         455   CALL CGENEW(XT,YT)
   1478              GO TO 1000
```

```
      456   CALL LAMPRE(XT,XT)
            GOTO 1000
      457   CALL LAMPRE(YT,YT)
            GOTO 1000
      458   CALL LAMPCP(XT,YT,XT,YT)
            GOTO 1000
      459   CALL CFAR(XT,XA)
            GOTO 1000
      460   CALL CFAR(YT,XB)
            GOTO 1000
      461   CALL DIGFIL(XT,YT)
            GOTO 1000
      462   CALL DIGFNC(XT,YT)
            GOTO 1000
      463   CALL DIGFSF(XT,YT)
            GOTO 1000
C
       501 CALL TARGET(XT,YT)
           GO TO 1000
       502 CALL TGTNCL(XT,YT)
           GO TO 1000
       503 IF(ICLCLN(26).EQ.0) GO TO 1121
           CALL CLUTTR (XT,YT,XR1,XR2,$1120)
           GO TO 1000
       504 CALL ANTPAT(XT,YT,$1060)                                        3540
           GO TO 1000
       505 CALL TSRPAT(XT,YT,XD1,XD2,XTM,YTM,$1162)
           GO TO 1000
      50    CALL PHDEC(XT,YT)
```

```
 CARD NO      ****                               CONTENTS

  1508              GOTO 1000
  1509        509   CALL FHDEC(XA,XB)
  1510              GOTO 1000
  1511        510   CALL WVGUID(XT,YT)
  1512              GOTO 1000
  1513        511   CALL IONOS(XT,YT)
  1514              GOTO 1000
  1515        512   CALL ECM(XT,YT)
  1516              GOTO 1000
  1517        C
  1518          107 CONTINUE
  1519          109 CONTINUE
  1520          112 CONTINUE
  1521          119 CONTINUE
  1522          120 CONTINUE
  1523        C
  1524          240 CONTINUE
  1525          241 CONTINUE
  1526          242 CONTINUE
  1527          243 CONTINUE
  1528        C
  1529          511 CONTINUE
  1530        C
  1531          421 CONTINUE
  1532          422 CONTINUE
  1533          424 CONTINUE
  1534          426 CONTINUE
  1535          429 CONTINUE
  1536          444 CONTINUE
```

```
C
  506 CONTINUE
  507   CONTINUE
  464   CONTINUE
  465   CONTINUE
  466   CONTINUE
  467   CONTINUE
  468   CONTINUE
  469   CONTINUE
  470   CONTINUE
C
      WRITE(6,50) JSTEP, MODULE
   50 FORMAT( ' NOTHING PERFORMED FOR ', I3, 'TH INPUT DATA ITEM. DATA I
     *TEM = ', I4 )
      GO TO 1000
C
  953 CONTINUE
      REWIND IFC
      WRITE(6,1953) IFC
 1953 FORMAT( ' TEMPORARY FILE NUMBER',I5,' HAS BEEN REWOUND')
      GO TO 1000
C
  954 CONTINUE
      WRITE(IFC) XTT
      IF( IT.GT.512) WRITE(IFC) (XT(J),J=513,IT )                        4140
      WRITE(6,1954) AXT,IFC
 1954 FORMAT( ' THE ARRAY',A6,'HAS BEEN STORED IN DATA SET NO.',I5)       4160
      GO TO 1000
C
```

```
        ****                      CONTENTS                              ****

          955 CONTINUE
              WRITE(IFC) YTT
              IF(ITY.GT.512) WRITE(IFC) (YT(J),J=513,ITY)                4210
              WRITE(6,1954) AYT,IFC
              GO TO 1000
        C
          956 CONTINUE
              WRITE(IFC) XAT
              IF(IXA.GT.512) WRITE(IFC) (XA(J),J=513,IXA)                4270
              WRITE(6,1954) AXA,IFC
              GO TO 1000
        C
          957 CONTINUE
              WRITE(IFC) XBT
              IF(IXB.GT.512) WRITE(IFC) (XB(J),J=513,IXB)                4330
              WRITE(6,1954) AXB,IFC
              GO TO 1000
        C
          958 CONTINUE
              READ(IFC) XTT
              IF( IT.GT.512)  READ(IFC) (XT(J),J=513,IT )                4390
              WRITE(6,1956) AXT,IFC
         1956 FORMAT( ' THE ARRAY',A6,'HAS BEEN LOADED FROM DATA SET NO.',I5)   4410
              GO TO 1000
        C
          959 CONTINUE
              READ(IFC) YTT
              IF(ITY.GT.512)  READ(IFC) (YT(J),J=513,ITY)                4460
              WRITE(6,1956) AYT,IFC
```

```
      GO TO 1000
C
  960 CONTINUE
      READ(IFC) XAT
      IF(IXA.GT.512)  READ(IFC) (XA(J),J=513,IXA)                    4520
      WRITE(6,1958) AXA,IFC
      GO TO 1000
C
  961 CONTINUE
      READ(IFC) XBT
      IF(IXB.GT.512)  READ(IFC) (XB(J),J=513,IXB)                    4580
      WRITE(6,1958) AXB,IFC
      GO TO 1000
  962 CONTINUE
      GO TO 1000
  963 CONTINUE
      GO TO 1000
  964 CONTINUE
      GO TO 1000
  965 CONTINUE
      GO TO 1000
  966 CONTINUE
      GO TO 1000
  967 CONTINUE
      GO TO 1000
  968 CONTINUE
      GO TO 1000
  969 CONTINUE
      GO TO 1000
  CARD NO        ****                        CONTENTS                                 ****

    1624         C
    1625           970 CONTINUE
    1626               DO 1970 K=1,8197
    1627          1970 XTT(K)=0.0
    1628                IT=2**N2
    1629                DELT=IT
    1630               WRITE(6,2970) AXT
    1631          2970 FORMAT(' THE ARRAY ',A6,' HAS BEEN SET TO 0.0 ' )                4850
    1632               GO TO 1000
    1633           971 CONTINUE
    1634               DO 1971 K=1,8197
    1635          1971 YTT(K)=0.0
    1636                ITY=2**N2
    1637                DELTY=IT
    1638               WRITE(6,2970) AYT
    1639               GO TO 1000
    1640           972 CONTINUE
    1641               DO 1972 K=1,8197
    1642          1972 XAT(K)=0.0
    1643               WRITE(6,2970) AXA
    1644               IF(MODULE.EQ.227) GO TO 1227
    1645               IF(MODULE.EQ.232) GO TO 1237
    1646                IXA=2**N2
    1647                XADEL=IT
    1648               GO TO 1000
    1649           973 CONTINUE
    1650               DO 1973 K=1,8197
    1651          1973 XET(K)=0.0
    1652               WRITE(6,2970) AXE
```

```
      IF(MODULE.EQ.227) GO TO 2227
      IF(MODULE.EQ.232) GO TO 2232
       IX6=2**N2
       XBDEL=11
      GO TO 1000
C
 1010 WRITE(6,1015)
 1015 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE WEIT' )
      GO TO 2020
C
 1020 WRITE(6,1025)                                                     5111
 1025 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE ISARY' )            5112
      GO TO 2020                                                        5113
C                                                                       5114
 1030 WRITE(6,1035)                                                     5141
 1035 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE FGENXY ')
      GO TO 2020
 1040  WRITE(6,1045)
 1045  FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE CGEN')
       GOTO 2020
C
 1050 WRITE(6,1055)
 1055 FORMAT( 1H , ' NONLINEAR TRANSFORM IMPROPERLY DEFINED ' )
      GO TO 2020
C
 1060 WRITE(6,1065)
 1065 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE ANTARY ')
      GO TO 2020
C
```

```
  1682          1070 WRITE(6,1075)
  1683          1075 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE PLTFMT')
  1684                GO TO 2020
  1685          C
  1686          1090 WRITE(6,1095)
  1687          1095 FORMAT( ' NON-STANDARD RETURN FROM SUBROUTINE SPCAVG ' )
  1688                GO TO 1000
  1689          C
  1690          1120 WRITE(6,1125)
  1691          1125 FORMAT(' NON-STANDARD RETURN FROM SUBROUTINE CLUTTR')
  1692                GO TO 1000
  1693          C
  1694          1121 WRITE(6,1122)
  1695          1122 FORMAT ('0THE CLUTTER MODEL HAS NOT BEEN PROPERLY INITIALIZED BY '
  1696               *'CLINT''.  ''CLUTTR'' WILL NOT BE EXECUTED.')
  1697                GO TO 2020
  1698          C
  1699          1130 WRITE(6,1135)
  1700          1135 FORMAT(' NON-STANDARD RETURN FROM SUBROUTINE CONV')
  1701                GO TO 2020
  1702          C
  1703          1182 WRITE(6,1183)
  1704          1183 FORMAT( ' SUBROUTINE TSRPAT NOT EXECUTED ')
  1705                GO TO 2020
  1706          C
  1707          2100 WRITE(6,2105) JSTEP, MODULE, ITEMP
  1708          2105 FORMAT( ' STEP ', I3, ' WITH MODULE = ', I4, ' HAS SCHEDULED AN OP
  1709               *ERATION WITH EXEC NUMBER ', I3,' WHICH IS OUT OF RANGE.' )
  1710                GO TO 1000
```

```
C
 2020  WRITE(6,2025)
 2025 FORMAT ('0A CONDITION EXISTS WHICH WILL CAUSE THE REMAINDER OF THI
     *S CONFIGURATION TO BE BYPASSED.')
       GO TO 290
C
 2050 WRITE(6,2055)
 2055 FORMAT( ' THIS JOB COMPLETE ' )
       CALL EXIT
 2060  WRITE(6,2065)
 2065  FORMAT(' THIS JOB ABORTED BECAUSE OF IMPROPER INPUT DATA',
     * '...SEE PRINT OUT FROM ACTIVITY 1')
      CALL EXIT
      STOP
      END
```

CALL DTOA (IX,X)

Where: IX contains the Input Waveform
X contains the Output Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. If the computed digital output is greater than the largest number which can be specified by NBITS, it will be set to the maximum number.

b. Flow Chart: Page 9-98

c. Cross Reference Table: 9-220.

6. THEORY OF OPERATION

The analog-to-digital conversion is performed by dividing the floating point input signal by the value of the least significant digital bit. If round off is to be performed, 0.5 is then added to the value. The value thus obtained is converted to an integer number and its absolute value is tested against the maximum allowable number, 2**(NBITS-1). If the magnitude of the signal is greater than the maximum allowable, it is set equal to the maximu. The basic mechanization equation for the ATOD module is:

IX (J) = IFIX(X(J)/XLSB + 0.5 * IROFF)

The digital-to-analog conversion is made by changing the input to a floating point number and multiplying it times the value of the least significant bit. The basic mechanization equation for the DTOA module is:

X(J) = FLOAT (IX(J)) * ZLSB

where: ZLSB = BOOL (IX(N196))

```
      SUBROUTINE ATOD(X,IX)
C
C**** THIS SUBROUTINE PERFORMS AN ANALOG TO DIGITAL (REAL TO INTEGER)
C          CONVERSION WITH SPECIFIED DIGITAL PRECISION AND SATURATION
C          LEVEL, OR A DIGITAL TO ANALOG (INTEGER TO REAL) CONVERSION
C          ON THE INPUT DATA ARRAY. *************************************
C
      COMMON/BLK1/BK1(500)
      DIMENSION X(1),IX(1)
      EQUIVALENCE  (BK1(103), XLSB        ),
     *            (BK1(104), NBITS       ), (BK1(105), IROFF       )
```

4-4a

```
 CARD NO        ****                              CONTENTS                              ****

  2010              *,            (BK1(149), ADCFS          )
  2011              DATA N193,N194,N195,N196/-3,-2,-1,0/
  2012              WRITE (6, 120)
  2013              TIADC=1.0/ABS(ADCFS)
  2014              T1=X(N195)
  2015              TIMADC=X(N194)
  2016              TIME=TIMADC+T1*0.5
  2017              N=IDUGL(X(N193))
  2018              K=1
  2019              IF (NBITS .LE. 31) GO TO 20
  2020              NBITS = 31
  2021              WRITE (6, 110) NBITS
  2022           20 CONTINUE
  2023              MAX = 2**(NBITS-1)
  2024              RCF = 0.5*FLOAT(IRCFF)
  2025              DXLSB = 1.0 / XLSB
  2026        C
  2027        C**** ANALOG TO DIGITAL CONVERSION ***************************************
  2028        C
  2029              DO 40  J = 1, N
  2030              IF(TIME.LT.TIMADC) GO TO 40
  2031              IX(K)=IFIX(X(J)*DXLSB+RCF)
  2032        C
  2033        C**** TEST FOR SATURATION ***********************************************
  2034        C
  2035              IF(IABS(IX(K)).GT.MAX) IX(K)=ISIGN(MAX,IX(K))
  2036              TIMADC=TIMADC+TIADC
  2037              K=K+1
                 40 CONTINUE
```

```
   30 IF(ADCFS) 31,32,32
   31 IX(K)=IX(K-1)
      K=K+1
   32 TIME=TIME+TI
   40 CONTINUE
C
      IX(N193)=K-1
      IX(N194) = IBOOL( X(N194) )
      IX(N195)=IBOOL(TIADC)
       IF(ADCFS.LT.0.0) IX(N195)=BOOL(TI)
      IX(N196) = IBOOL( XLSB )
      RETURN
      ENTRY DTOA(IX,X)
      WRITE (6, 130)
      X(N193) = BOOL(IX(N193))
      X(N194) = BOOL(IX(N194))
      X(N195) = BOOL(IX(N195))
      N = IX(N193)
      ZLSB=BOOL(IX(N196))
C
C**** DIGITAL TO ANALOG CONVERSION *************************************
C
      DO 60 J= 1, N
      X(J) = FLOAT(IX(J)) * ZLSB
   60 CONTINUE
      RETURN
  110 FORMAT (' NBITS IS EXCESSIVE.  THE VALUE OF NBITS HAS BEEN SET TO
     +',I  )
  120 FORMAT(/// 49X,'* * * * A TO D CONVERTER * * * *'/ / )
  130 FORMAT(/// 49X,'* * * * D TO A CONVERTER * * * *' / / )
      END
```

SUBROUTINE CDIGFL

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CDIGFL	LTI-4	403
CDFNCL	LTI-4	404

2. PURPOSE:

This subroutine is used to simulate a cross-coupled digital filter and synthesize a filter transfer function having one complex pole and/or one complex zero.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FFR	0	F	Feed - Forward coefficient - R
FFI	0	F	Feed - Forward coefficient - I
FBR	0	F	Feedback coefficient - R
FB1	0	F	Feedback coefficient - I

4. CALLING SEQUENCES:

CALL CDIGFL (X,Y)

Where:

X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage registers (TX1 and TY1) are cleared before execution begins.

CALL CDFNCL (X,Y)

Where:

X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage registers, TX1 and TY1, are not cleared before execution begins.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-96

b. Cross Reference Table: 9-220

6. THEORY OF OPERATION

The block diagram of the complex digital filter simulated by this module is shown in Figure CDIGFL-1. The Z-plane transfer function is given by the following expression:

$$T(Z) = \frac{Z - (FFR + jFFI)}{Z - (FBR + jFBI)}$$

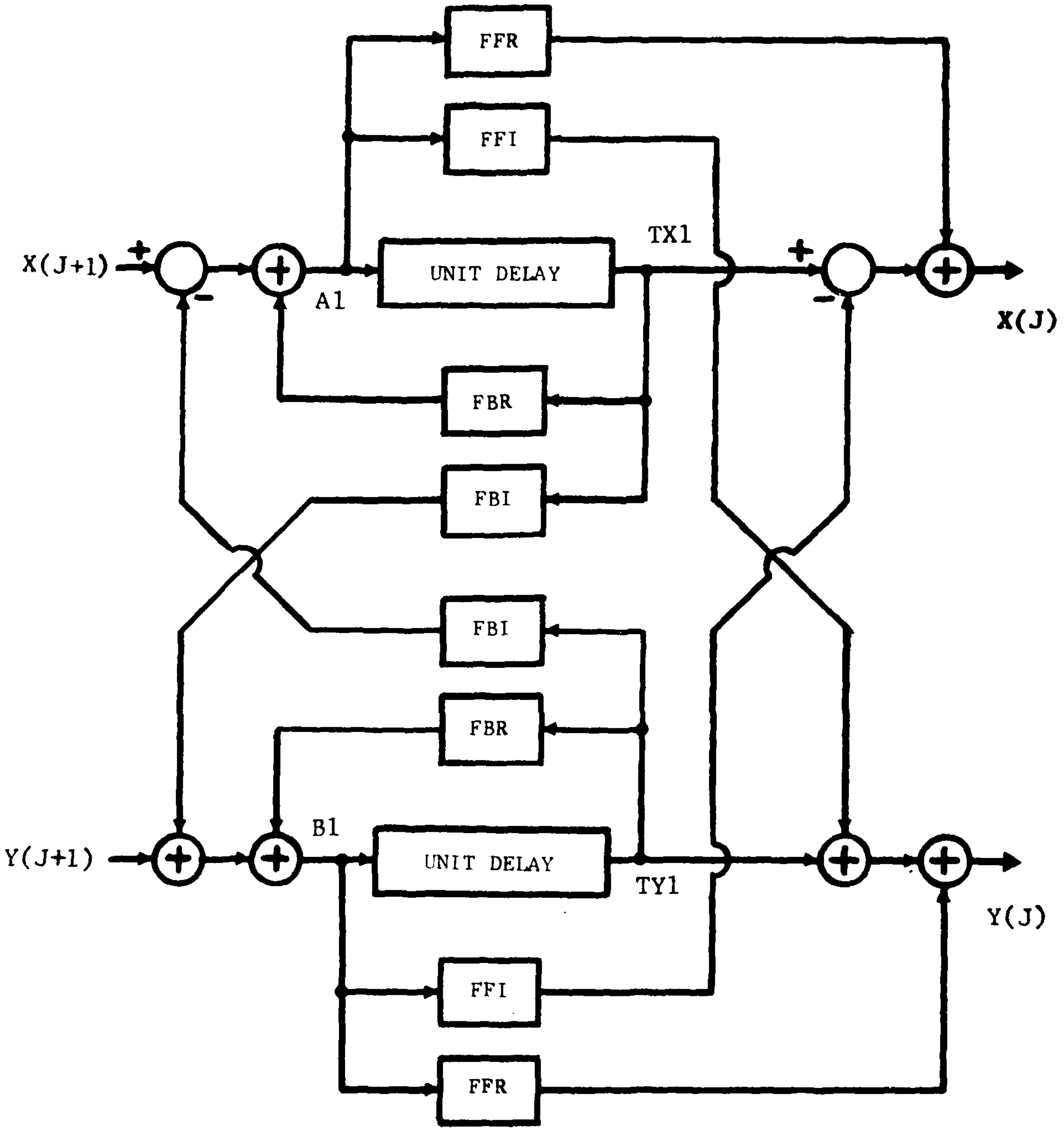


Figure CDIGFL-1 BLOCK DIAGRAM OF CDIGFL/CDFNCL

```
      SUBROUTINE CDIGFL(X,Y)
      COMMON/BK1/BK1(500)
      EQUIVALENCE (BK1(68), FFX),(BK1(69), FFXY),                   470
     *            (BK1(70), FBX),(BK1(71), FBXY)                    480
      DATA N195/-5/                                                 490
      DIMENSION X(1),Y(1)
      TX1=0.0                                                       510
      TY1=0.0                                                       520
      ENTRY CDFNCL(X,Y)
      N=DECDE(X(N195))
      DO 10 J=1,N                                                   570
      A1=X(J)+TX1*FBX-TY1*FBXY
      B1=Y(J)+TY1*FBX+TX1*FBXY
      X(J)=A1*FFX-B1*FFXY+TX1
      Y(J)=B1*FFX+A1*FFXY+TY1
      TX1=A1                                                        620
      TY1=B1                                                        621
   10 CONTINUE                                                      622
      RETURN                                                        623
      END                                                           624
```

SUBROUTINE CFAR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CFAR	NL-2	459,460

2. PURPOSE:

This subroutine is used to simulate a constant false alarm rate (CFAR) video processor.

3. INPUT PARAMETERS:

Name	O/R	T	Description
TAVG	R	F	Width of averaging window used in determining video gain (TAVG ≥ 2*TI)

4. CALLING SEQUENCES:

CALL CFAR(VIN,VOUT)

Where: VIN contains the Input Waveform

VOUT contains the Output Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The input and output arrays cannot be equivalenced.

b. Flow Chart: Page 9-66

c. Cross Reference Table: Page 9-216

6. THEORY OF OPERATION

The block diagram of the CFAR processor simulated by this module is shown in Figure CFAR-1. Each sample of the input waveform is passed through a variable gain amplifier which is controlled by the average value

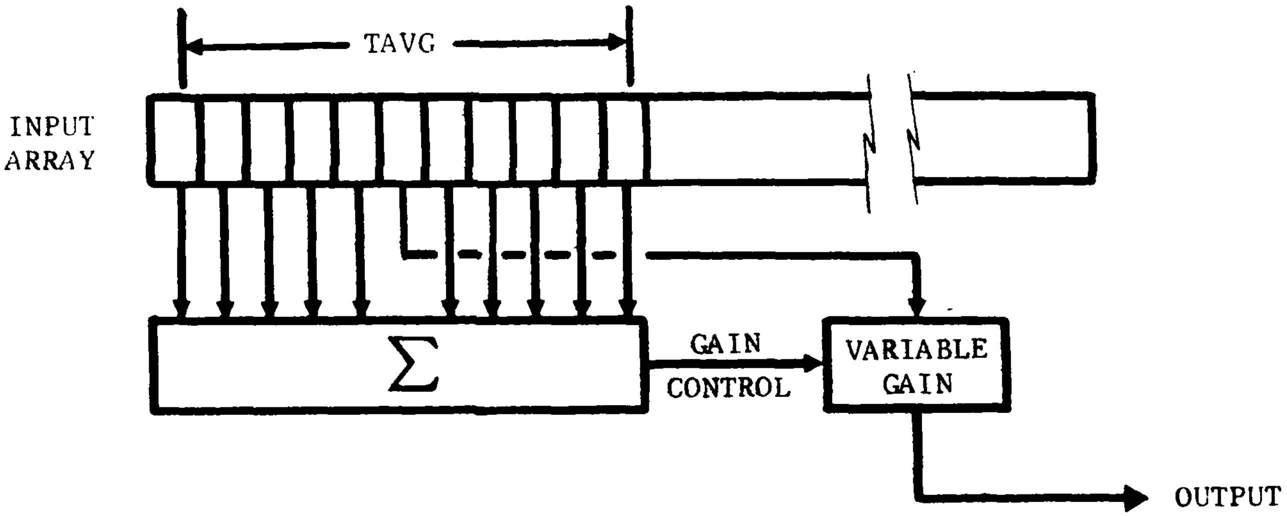


Figure CFAR-1 BLOCK DIAGRAM OF CFAR

of this waveform surrounding the sample being processed. The gain applied to the Jth sample is given by the following expression:

$$G(J) = \frac{2*NCELL2}{\sum_{K = J-NCELL2}^{K = J+NCELL2} \left(\left| VIN(K) \right| - \left| VIN(J) \right| \right)}$$

where: $NCELL2 < J \leq NPTS - NCELL2$

$NCELL2 = IFIX(TAVG*0.5/TI)$

TI = independent variable increment between samples

NPTS = number of samples in the input array

```
      SUBROUTINE CFAR(VIN,VOUT)
      COMMON/BLK1/ BK1(200)
      DIMENSION VIN(1),VOUT(1)
      EQUIVALENCE (BK1(196), TAVG )
      DATA N193,N194,N195/-3,-2,-1/
      NCELL2=IFIX(TAVG*0.5/VIN(N195))
      XCELL=2*NCELL2
      NPTS=BOOL(VIN(N193))
      NSTOP=NPTS-NCELL2
      NCELL1=NCELL2+1
      AV=0.0
      DO 100 J=1,NCELL2
      AV=AV+ABS(VIN(J))
  100 CONTINUE
      GF=1.0
      DO 200 J=1,NPTS
      IF(J.LE.NSTOP) AV=AV+ABS(VIN(J+NCELL2))
      IF(J.GT.NCELL1) AV=AV-ABS(VIN(J-NCELL1))
      DVSR=AV-ABS(VIN(J))
      IF(DVSR.GT.1.0E-20) GF=XCELL/DVSR
      VOUT(J)=VIN(J)*GF
  200 CONTINUE
      VOUT(N193)=VIN(N193)
      VOUT(N194)=VIN(N194)
      VOUT(N195)=VIN(N195)
      RETURN
      END
```

SUBROUTINE CLUTTR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CLUTTR	LTI-3 or LTV-3	503

2. PURPOSE:

This subroutine generates the transfer function of a clutter volume. A pictorial diagram of the clutter volume is shown in Figure CLUTTR-1.

3. INPUT PARAMETERS:

All required input data is entered via the clutter model initialization subroutine (CLINT).

4. CALLING SEQUENCES:

CALL CLUTTR(XT,YT,GAZ,GEL)

WHERE: XT contains the Output Impulse response - R

YT contains the Output Impulse response - I

GAZ-Temporary storage for antenna azimuth gain values (MM=number of storage cells required)

GEL-Temporary storage for antenna elevation gain values (NN - number of storage cells required)

5. RESTRICTIONS, REQUIREMENTS, AND MISCELLANEOUS DATA

a. The subroutines CLINT must have been successfully executed prior to calling CLUTTR. See p 6-4 for restrictions on CLINT.

b. The functions AZGAIN and ELGAIN are used to compute the antenna gain associated with each scatterer.

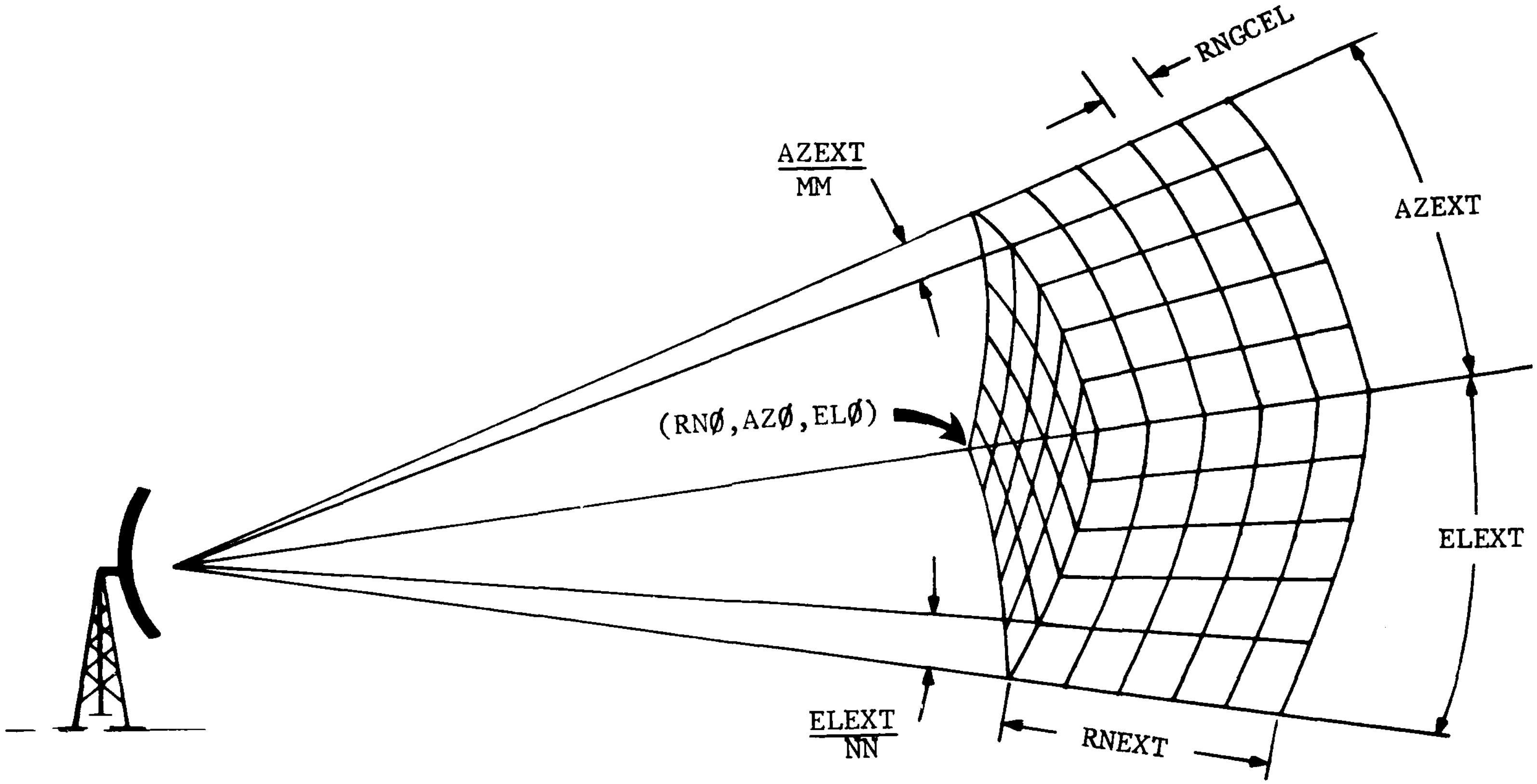


Figure CLUTTR-1 CLUTTER MODEL GEOMETRY

c. Flow Chart: Page 9-112

d. Cross Reference Table: Page 9-222

6. THEORY OF OPERATION

The clutter model is based on the premise that a volume of clutter can be represented by an ensemble of discrete scatterers. The RCS of these scatterers is derived from a theoretical probability density function, but can be easily modified to allow the use of a density function derived from measurements data. The phase behavior of the scatterer is determined from motion of the scatterer due to wind and from any inherent frequency spread derived from measurements data. The basic mechanization equation for the clutter model is given by the following expressions:

a. MODE = 1 (stationary clutter):

$$XT(K) + jYT(K) = \sum_{N=1}^{NN} GEL(N) \sum_{M=1}^{MM} GAZ(M) \left| CLUX(IPNT) + jCLUY(IPNT) \right|$$

where: GEL(N) - Normalized antenna gain corresponding to an elevation angle of ELØØØ + DELEL *(N-1)

GAZ(M) - Antenna gain corresponding to an azimuth angle of AZØØØ + DELAZ * (M-1)

IPNT - Scatterer element pointer which is determined as follows:
IPNT = (K*(N-1)(M-1)+(N-1)*MM+M) MODULO 250

CLUX( )- Array containing real part of clutter scatterer specification

CLUY( )- Array containing imaginary part of clutter scatterer specification

b. MODE = 2 (Time varying clutter):

$$XT(K)+jYT(K) = \sum_{N=1}^{NN} GEL(N) \sum_{M=1}^{MM} GAZ(M) \underbrace{\left| CLUX(PDNT)+jCLUY(IDNT) \right| * e^{PHDEL}}_{\text{"New" } CLUX(IDNT)+jCLUY(PDNT)}$$

Where: PHDEL = is the phase term to be applied to each scatter

$$= 57.29578*\mathrm{DELTIM}*\mathrm{DOPFRQ}*\mathrm{COS}\left\{\left[\mathrm{XVLANG}-\mathrm{DELAZ}*(M-1)\right]/57.29567\right\} + \mathrm{PHRW}$$

DELTIM = Radar interpulse period

PHRW = The random walk phase. This variable is generated by a random number generator from a umiform distribution.

$$\frac{-\mathrm{RWPH}}{2.0} < \mathrm{PHRW} < \frac{\mathrm{RWPH}}{2.0}$$

The pulse phase shift due to doppler is computed via a recursive loop which is more efficient than a direct evaluation of the cosine function.

$$C(N+1) = C(N)*\mathrm{DELCS} - S(N)*\mathrm{DELSN}$$
$$S(N+1) = C(N)*\mathrm{DELSN}+S(N)*\mathrm{DELCS}$$

where C = Phase shift due to doppler

$$C(1) = \mathrm{DOPFRQ}*\mathrm{DELTIM}*2\pi * \mathrm{COS}(\mathrm{XVLANG})$$
$$S(1) = \mathrm{DOPFRQ}*\mathrm{DELTIM}*2\pi * \mathrm{SIN}(\mathrm{XVLANG})$$

The clutter scatterer parameters (CLUX and CLUY) are updated by PHDEL for each execution of the clutter model and the new values are stored in the random disc file.

AD-A102 278   GENERAL DYNAMICS FORT WORTH TEX CONVAIR AEROSPACE DIV   F/6 17/9
ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING. VOLUME II. PAR--ETC(U)
JUN 76 R J HANCOCK, F H CLEVELAND   F30602-73-C-0380
UNCLASSIFIED   RADC-TR-76-186-VOL-2-PT-1   NL

2 of 4
AD
A102278

```
      SUBROUTINE CLUTTR(XT,YT,GAZ,GEL,*)
      COMMON/BLK1/ BK1(200)
      COMMON/BLK330/BK4(30)
       DIMENSION GAZ(1),GEL(1),CLUX(250),CLUY(250)
      COMMON/BLKRND/ IIII(12),NRAND(129)
      DIMENSION XT(1),YT(1),CSCAT(500),IRAND(128)
      EQUIVALENCE (CLUX(1),CSCAT(1)) , (IRAND(1),NRAND(2))
       EQUIVALENCE (CLUY(1),CSCAT(251))
      EQUIVALENCE  (BK1( 14), TCELL        ),(BK1( 21), IOMP        ),
     *             (BK1( 17), TI           ),(BK1(  2), FS          ),
     *             (BK4(  3), RWPH         ),
     *             (BK4(  6), RNEXT        ),
     *             (BK4(  7), RNOOO        ),(BK4(  9), AZOOO       ),
     *             (BK4( 10), MM           ),(BK4( 12), ELOOO       ),
     *             (BK4( 13), NN           ),(BK4( 14), KK          ),
     *             (BK4( 16), MODE         ),(BK4( 17), DELAZ       ),
     *             (BK4( 18), DELEL        ),(BK4( 19), XVLANG      ),
     *             (BK4( 20), ICFLG        ),(BK4( 21), DOPPFQ      )
     * ,           (BK4( 22), KCELL        )
      EQUIVALENCE  (BK1( 16), TIME         )
      DATA N193,N194,N195,N196/-3,-2,-1,0/
       DATA OTIME,IMULT,F12/0.0,1220703125,6.2831853/
       CALL RANSIZ(02,500)
C
      WRITE(6,10) (BK4(J),J=1,30)
   10 FORMAT(1H , O12,8X,O12,8X,O12,8X,O12,8X,O12)
       RWPHC= RWPH*F12/( 360.0*(2.0**35))
       DELAZR=DELAZ/57.29578
      DELTIM= TIME-OTIME
      OTIME=TIME
      D= DOPPFQ*DELTIM*F12
      DL=D*COS(XVLANG-DELAZR)
```

CLO
4-18a

```
      SN=D*SIN(ZVLANG-DELAZK)
      DELCS=COS(DELAZK)
      DELSN=SIN(DELAZK)
      ANGLE=AZ000
      DO 100 M=1,MM
      CAZ(M)=AZCAIN(ANGLE)
      ANGLE=ANGLE+DELAZ
  100 CONTINUE
      IF(IDMP.NE.0) WRITE(6,1001) (CAZ(J),J=1,MM)
      ANGLE=EL000
      DO 200 N=1,NN
      GEL(N)=ELCAIN(ANGLE)
      ANGLE=ANGLE+DELEL
  200 CONTINUE
      IF(IDMP.NE.0) WRITE(6,1001) (GEL(J),J=1,NN)
      DO 50 J=1,KCELL
      XI(J)=0.0
      YI(J)=0.0
   50 CONTINUE
C
      RTIME=0.5*TI
      IREC=1
      TAKW=0.0
      IFNT=1
      READ(2'IREC) CSCAT
      DO 500 K=1,KK
      A=0.0
      B=0.0
      DO 600 N=1,NN
```

```
CARD NO        ****                          CONTENTS                          ****

 3016              C=CS
 3017              S=SN
 3018              DO 700 M=1,MM
 3019              IF(IPNT.LE.250) GO TO 750
 3020              GO TO(725,730),MODE
 3021        730    WRITE(2'IREC) CSCAT
 3022        725    IREC=IREC+1
 3023              READ(2'IREC) CSCAT
 3024              IPNT=1
 3025         750 G=GAZ(M)*GEL(N)
 3026              GO TO(760,770),MODE
 3027         760 A=A+CLOX(IPNT)*G
 3028              B=B+CLOY(IPNT)*G
 3029              GO TO 780
 3030         770 CONTINUE
 3031                IF(KWPH.EQ.0.0) GOTO 780
 3032              IRND=IRAND(MAD)
 3033              IRND=IRND+JRND*IMULT
 3034                IRND=FLD(1,35,IRND)
 3035                JRND=IRND
 3036                IRAND(MAD)=IRND
 3037                MAD=FLD(15,7,IRND)
 3038                PHEW=FLOAT(IRND)*KWPHC
 3039          780 CONTINUE
 3040               IF(C.EQ.0.0) GO TO 790
 3041               TIMP=C
 3042               C=C*DELCS-S*DELSN
 3043               S=TIMP*DELSN+S*DELCS
 3044          790 CONTINUE
```

```
      FHCEL=FHRW*G
      VCOS=COS(FHCEL)
      VSIN=SIN(FHCEL)
      TIMP=CLOX(IPNT)
      CLOX(IPNT)=TIMP*VCOS-CLOY(IPNT)*VSIN
      CLOY(IPNT)=TIMP*VSIN+CLOY(IPNT)*VCOS
      A=A+G*CLOX(IPNT)
      B=B+G*CLOY(IPNT)
  800 CONTINUE
      IPNT=IPNT+1
  700 CONTINUE
  600 CONTINUE
      LOC=IFIX(KTIME*FS)
      XT(LOC)=A
      YT(LOC)=B
      KTIME=KTIME+ICELL
  500 CONTINUE
C
      IF(MODE.EQ.2) WRITE(2'IREC) CSCAT
      XT(N193)=FLOAT(KCELL)
      XT(N194)=KNODO
      XT(N195)=T1
      YT(N193)=XT(N193)
      YT(N194)=XT(N194)
      YT(N195)=XT(N195)
 1001 FORMAT(1H ,6E20.5)
       IF(IDMP.EQ.1) WRITE(6,1001) (XT(J), YT(J), J=1,KCELL )
      RETURN
      END
```

SUBROUTINE DCFAR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
DCFAR	NL-2	440,441

2. PURPOSE:

This subroutine is used to simulate a digital constant false alarm rate (CFAR) video processor.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NCELL	R	I	Number of cells to be averaged in computing the video gain.

4. CALLING SEQUENCES:

CALL DCFAR (IN, IOUT)

Where: IN contains the Input Waveform
IOUT contains the Output Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The input and output arrays cannot be equivalenced.

b. All calculations are in integer format.

c. Flow Chart: Page 9-93

d. Cross Reference Table: Page 9-220

6. THEORY OF OPERATION

The block diagram of the digital CFAR processor simulated by this module is shown in Figure DCFAR-1. Each sample of the input waveform is passed through a variable gain amplifier which is contro led by the average value of the waveform surrounding the sample being processed. The gain applied to the Jth sample is given by the following expression:

$$IG = \frac{NCELL}{\sum_{K=J-NCELL2}^{K=J+NCELL2} \left|IN(K)\right| - \left|IN(J)\right|}$$

where: $NCELL\ 2 < J \leq NPTS - NCELL2$

$NCELL\ 2 = NCELL/2$

NPTS = Number of input samples in the input array

The value of the Jth output sample is computed as follows:

$$IOUT(J) = IN(J) * IG$$

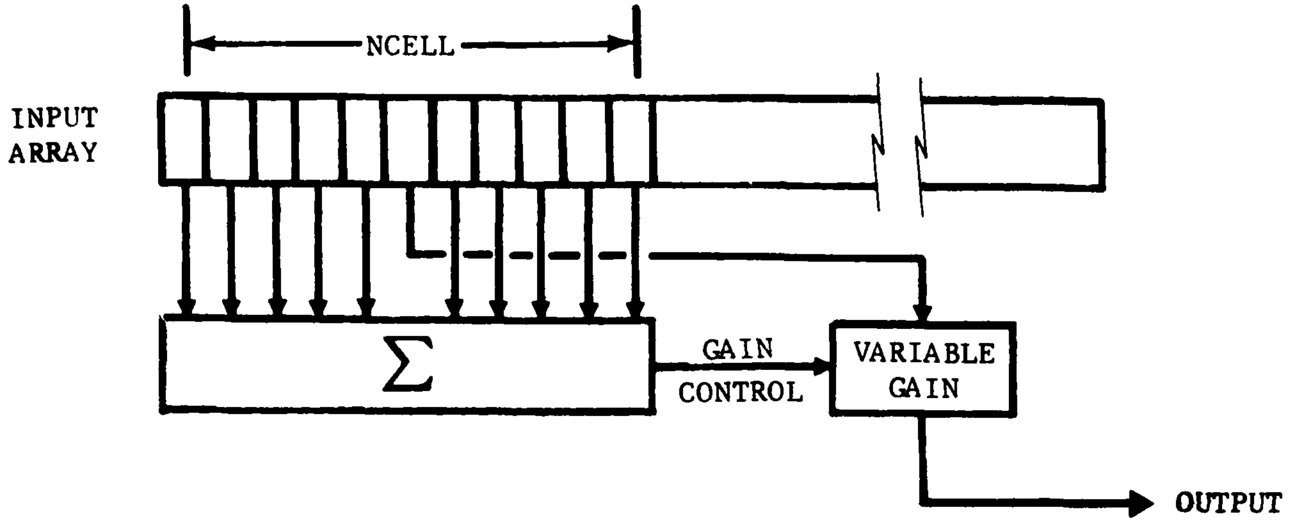


Figure DCFAR-1 BLOCK DIAGRAM OF DCFAR

```
      SUBROUTINE DCFAK(IN,IOUT)
      COMMON/GEN1/ GN1(200)
      DIMENSION IN(1),IOUT(1)
      EQUIVALENCE  (GN1(170), NCELL
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      NCELL2=NCELL/2
      NCELL=NCELL2*2
      NPTS=IN(N193)
      NSTOP=NPTS-NCELL2
      IAV=0
      DO 100 J=1,NCELL2
      IAV=IAV+IN(J)
  100 CONTINUE
      NCELL1=NCELL2+1
      DO 200 J=1,NPTS
      IF(J.LE.NSTOP) IAV=IAV+IABS(IN(J+NCELL2))
      IF(J.GT.NCELL1)IAV=IAV-IABS(IN(J-NCELL1))
      ICF=IAV-IABS(IN(J))
      IF(ICF.EQ.0) ICF=1
      IOUT(J)>(IN(J)*NCELL)/ICF
  200 CONTINUE
      IOUT(N193)=IN(N193)
      IOUT(N194)=IN(N194)
      IOUT(N195)=IN(N195)
      IOUT(N196)-IN(N196)
      RETURN
      END
```

SUBROUTINE DFT

1. **MODULE IDENTIFICATION:**

Name	Classification Code	Reference Number
DFT	LTI-2	201
DFTFØ	LTI-2	234
DFTRF	LTI-2	233
DFTNCL	LTI-2S	None

2. **PURPOSE:**

This subroutine computes the finite discrete Fourier transform of a sequence of δ-functions.

3. **INPUT PARAMETERS:**

Name	O/R	T	Description
SIMFØ	0	F	Center frequency of discrete output spectrum (DFTFØ only)
FI	R	F	Frequency increment between samples of the output spectrum
FEXT	R	F	Width of output spectrum
NIMP	R	I	Number of δ-functions to be transformed
IDMP	R	I	Diagnostic data dump parameter = 0; no diagnostic data = 1; dump internal parameters: PS, DELPS, NPTS, NIMP, DIN = 2; dump output spectrum: PS, DELPS, NPTS, NIMP, DIN
FØ	0	F	Center frequency of discrete output spectrum (DFTRF only)
INORM	R	I	Normalization Flag = 0; f_N = TI = 1; f_N = 1.0/NIMP = 2; f_N = TI = 3; f_N = 1.0/NIMP = 4; f_N = 1.0

Name	O/R	T	Description
TI	O	F	Time increment between simulated waveform samples. Used in this subroutine for normalization only (required only if INORM = 0 or 2)
DIN	R	F	Array containing the parameters of the δ-functions. The specification for the Jth δ-function is the following: DIN(1,J) = phase angle DIN(2,J) = time of occurrence DIN(3,J) = amplitude

4. CALLING SEQUENCES:

Video spectrum

CALL DFT (X,Y)

Where: X contains the Output Waveform - R

Y contains the Output Waveform - I

The range of the independent variable, f, used in computing the spectrum is:

$$-\frac{\mathrm{FEXT}}{2} \le f < \frac{\mathrm{FEXT}}{2}$$

IF spectrum

CALL DFTFØ (X,Y)

Where: X contains the Output Waveform - R

Y contains the Output Waveform - I

The range of the independent variable, f, used in computing the spectrum is:

$$\mathrm{SIMF\emptyset} - \frac{\mathrm{FEXT}}{2} \le f < \mathrm{SIMF\emptyset} + \frac{\mathrm{FEXT}}{2}$$

RF spectrum

CALL DFTRF (X,Y)

where: X contains the Output Waveform - R

Y contains the Output Waveform - I

The range of the independent variable, f, used in computing the spectrum is:

$$F\emptyset - \frac{FEXT}{2} \leq f < F\emptyset + \frac{FEXT}{2}$$

Transform additional sample

CALL DFTNCL (X,Y)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

The computed spectrum is added to the spectrum contained in arrays X and Y. The range of the independent variable is determined by the previous execution of DFT, DFTFØ, or DFTRF.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. This subroutine is quite time consuming and therefore should be used only when necessary. For most applications the AFFT subroutine which uses the Fast Fourier Transform algorithm should be used. The execution time for this subroutine can be estimated using the following expression

$$\frac{FEXT}{FI} \times NIMP \times 50\ \mu S$$

b. This subroutine is used normally in conjunction with other subroutines such as PHDEC and CGEN.

c. The output spectrum has a center frequency of zero, i.e. it is the low pass equivalent.

d. Flow Chart: Page 9-142

e. Cross Reference Table: Page 9-226

f. DFT lets user generate 100 delays to be considered.

6. THEORY OF OrERATION

The Fourier transform of a sequence of samples represented by S-functions is given by the following expression:

$$S(f) = F_N \int_{-\infty}^{\infty} \left[\sum_{m=1}^{NIMP} a(m) \quad e^{j\emptyset(m)} \; \delta(t-t_{(m)}) \right] e^{-j2\pi ft} \quad dt$$

where: F_N is the normalization factor determined by the flag INORM.

Interchanging the order of integration and summation the following result is obtained:

$$S(f) = F_N \sum_{m=1}^{NIMP} a_{(m)} e^{j\emptyset(m)} \; e^{-j2\pi ft(m)}$$

If only K equally spaced samples of S(f) are computed, the following expression results:

$$S_k = S_\star(f) \Big|_{f=kFI+f_o} = F_N \sum_{m=1}^{NIMP} a(m) \; e^{j\emptyset(m)} \; e^{-j2\pi kFIt(m)}$$

where: $K = \frac{FEXT}{FI}$ and f_o is defined in the following manner for each module name:

DFT: $f_o = -\frac{FEXT}{2}$

DFTFØ: $f_o = SIMF\emptyset - \frac{FEXT}{2}$

DFTRF: $f_o = F\emptyset - \frac{FEXT}{2}$

```
      SUBROUTINE DFT(X,Y)
      COMMON/BLK1/ BK1(200),DIN(3,100)
      EQUIVALENCE (SIMFO,BK1(8)) , (F1,BK1(11)) , (FEXT,BK1(4)),
     *             (NIMP,BK1(200)), (IDMP,BK1(21)), (FO,BK1(3))
     * ,(INCRM ,BK1(9)), ( T1  ,BK1(12))
      DIMENSION     X(1) ,  Y(1)
      DATA N193,N194,N195,PI2,D1/-3,-2,-1,6.2831853,2.7777778E-03/         1650
       FCENT=0.0
       GOTO 10
C
       ENTRY DFTFO(X,Y)
      FCENT=SIMFO
      GO TO 10
C
      ENTRY DFTRI(X,Y)
```

```
          FCENT=F0
C
       10 CONTINUE
          NPTS = IFIX(FEXT / FI )
          NPTS2= NPTS/2
          NPTS= 2*NPTS2
          DO 50 J=1,NPTS
          X(J)=0.0
          Y(J)=0.0
       50 CONTINUE
          X(N193)=EUUL(NPTS)
          X(N194)=-FI*FLOAT(NPTS2)
          X(N195)=FI
          Y(N193)=X(N193)
          Y(N194)=X(N194)
          Y(N195)=X(N195)
C
          ENTRY DFTNCL(X,Y)
          IF(IDMP) 60,60,55
       55 CONTINUE
          WRITE(6,1007) ((DIN(K,J),K=1,3),J=1,NIMP)
     1007 FORMAT(1H ,6E15.5)
       60 CONTINUE
           XN=FI
           IF(INORM.EQ.1.OR.INORM.EQ.3) XN=1.0/FLOAT(NIMP)
           IF(INORM.EQ.4) XN=1.0
          DO 200 J=1,NIMP
          A=DIN(3,J)*XN
          T=DIN(2,J)
```

```
      PH=-PCON(I+1)*SIN(I,J)*DT
      DELPH=-PI*4
      IPASS=1
      K=NPTS2+1
      KDEL=1
C
   60 CONTINUE
      PS=(PH-AINT(PH))*PI2
      DELPS=(DELPH-AINT(DELPH))*PI2
      CS= COS(PS)*A
      SN= SIN(PS)*A
      ACDEL=COS(DELPS)
      ASDEL=SIN(DELPS)
      IF(ICMP.EQ.0) GO TO 70
      WRITE(6,1002)PS,DELPS,NPTS,NIMP
 1002 FORMAT(' PS=',1PE13.6,'   DELPH=',1PE13.6,'   NPTS=',I10,'   NIMP=
     +',I10)
   70 CONTINUE
      DO 400 L=1,NPTS2
      X(K)=X(K)+CS
      Y(K)=Y(K)+SN
      TEMP=CS
      CS=CS*ACDEL-SN*ASDEL
      SN=SN*ACDEL+TEMP*ASDEL
      K=K+KDEL
  400 CONTINUE
C
      IF(IPASS.EQ.2) GO TO 200
      IPASS=2
```

4-29a

```
CARD NO.        ****                    CONTENTS                     ****

                      K=NPTS2
                      KDEL=-1
                      PH=PH-DELPH
                      DELPH=-DELPH
                      GO TO 71
                  200 CONTINUE
                C
                      IF(ILMF-2) 80,75,80                              2430
                   75 CONTINUE
                      WRITE(6,1002) (X(M),Y(M), M=1,NPTS)
                 1002 FORMAT(1H0, 6E20.5)
                   80 CONTINUE
                C
                      RETURN
                      END
```

SUBROUTINE DIGFIL

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
DIGFIL	LTI-4	461
DIGFNC	LTI-4	462
ECMFL	LTI-4	None

2. PURPOSE:

This subroutine is used to simulate a multiple section digital filter.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NP	R	I	Number of 2-delay sections
SF	R	F	Scale factor
FB	0	F	Array containing the feedback coefficients. The coefficients for the Kth section are the following: FB(1,K) = 1-delay feedback coef. FB(2,K) = 2-delay feedback coef.
FF	0	F	Array containing the feed-forward coefficients. The coefficients for the Kth section are the following: FF(1,K) = 0-delay feed-forward coef. FF(2,K) = 1-delay feed-forward coef.

4. CALLING SEQUENCES:

CALL DIGFIL (X,Y)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage register arrays (XM and YM) are cleared before execution begins.

CALL DIGFNC (X,Y)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage register arrays (XM and YM) are not cleared before execution begins.

CALL ECMFL (X,Y)

Where: X contains the Input Waveform sample - R
Y contains the Input Waveform sample - I
X contains the Output Waveform sample - R
Y contains the Output Waveform sample - I

One complex sample processed for each execution of the module.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-101
b. Cross Reference Table: Page 9-221

6. THEORY OF OPERATION

The block diagram of one section of the digital filter simulated by this module is shown in Figure DIGFIL-1. The Z-plane transfer function for this section is given by the following expression:

$$T_K(Z) = FF(1,K)\,\frac{Z^2 + \frac{FF(2,K)}{FF(1,K)}\,Z + \frac{1}{FF(1,K)}}{Z^2 - FB(1,K)\,Z - FB(2,K)}$$

The Z-plane transfer function for the complete filter is given by the following expression:

$$T(Z) = SF \prod_{K=1}^{K=NP} T_K(Z)$$

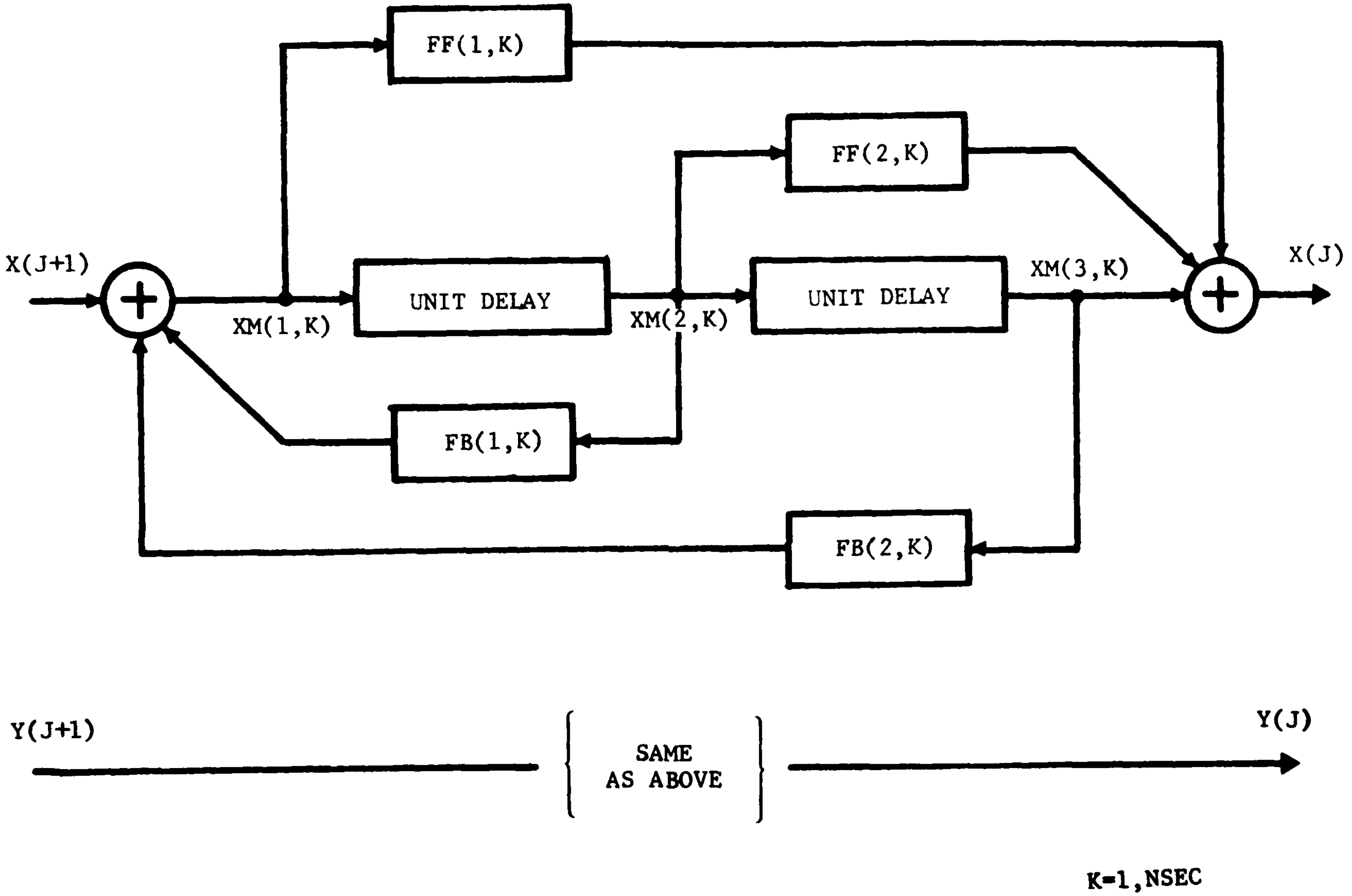


Figure DIGFIL-1 BLOCK DIAGRAM OF DIGFIL/DIGFNC

```
      SUBROUTINE DIGFIL(X,Y)
      COMMON/BLK1/ BK1(200),FB(2,25),FF(2,25)
      EQUIVALENCE (BK1(199), NP ) , (BK1( 74), SF )
      DATA N193/-3/
      DIMENSION X(1),Y(1),XM(3,25),YM(3,25)
      DO 100 J=1,25
      DO 100 K=1,3
      XM(K,J)=0.0
      YM(K,J)=0.0
  100 CONTINUE
      ENTRY DIGFNC(X,Y)
      N=ALCUL(X(N193))
      GOTO 150
      ENTRY ECMFL(X,Y)
      N=1
  150 DO 200 J=1,N
      XX=X(J)
      YY=Y(J)
      DO 300 K=1,NP
      XM(1,K)=XM(2,K)*FB(1,K)+XM(3,K)*FB(2,K)+XX
      YM(1,K)=YM(2,K)*FB(1,K)+YM(3,K)*FB(2,K)+YY
      XX=XM(1,K)*FF(1,K)+XM(2,K)*FF(2,K)+XM(3,K)
      XY=YM(1,K)*FF(1,K)+YM(2,K)*FF(2,K)+YM(3,K)
      XM(3,K)=XM(2,K)
      XM(2,K)=XM(1,K)
      YM(3,K)=YM(2,K)
      YM(2,K)=YM(1,K)
  300 CONTINUE
      X(J)=XX*SF
      Y(J)=YY*SF
  200 CONTINUE
      RETURN
      END
```

SUBROUTINE DIGFSF

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
DIGFSF	LTI-4	463
ECMFSU	LTI-4	None
ECMFSF	LTI-4	None

2. PURPOSE:

This subroutine is used to simulate a digital filter based on the frequency sampling design concept.

3. INPUT PARAMETERS:

Name	O/R	T	Description
RADIUS	R	F	Radius of circle in Z-plane which is locus of pole/zero locations
NSAM	R	I	Number of frequency samples used to specify the transfer function
NDFZ	R	I	Number of zeros synthesized
FSAM	R	F	Array containing the filter transfer function specification. The coefficients for the Kth sample are the following: FSAM (1,K) = real coefficient FSAM (2,K) = imaginary coefficient The frequency associated with the Kth sample is given by the following expression. $f_K = \frac{fs}{NDFZ}(K-NDC)$ K = 1, NSAM where: NDC = IFIX (NSAM/2) + 1 f_s = Sampling rate of data to be processed

4. CALLING SEQUENCES:

Process a waveform

CALL DIGFSF (X,Y)

Where:
X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage register arrays (S and PF) are cleared before execution begins.

Set up for use by ECM module

CALL ECMFSU (X,Y)

Where:
X contains the Input Waveform sample - R
Y contains the Input Waveform sample - I
X contains the Output Waveform sample - R
Y contains the Output Waveform sample - I

The storage register arrays (S and PF) are cleared and coefficients are initialized.

Process 1 sample (called by ECM module)

CALL ECMFSF (X,Y)

Where:
X contains the Input Waveform sample - R
Y contains the Input Waveform sample - I
X contains the Output Waveform sample - R
Y contains the Output Waveform sample - I

The storage register arrays (S and PF) are not cleared before execution begins.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. This subroutine is structured to simulate a low pass digital filter only.

b. *References:*

C. M. Rader and B. Gold: "Digital Filter Design Techniques in the Frequency Domain", Proc. IEEE, Vol. 55, pp 149-171, Feb. 1967.

L. R. Rabiner, B. Gold, and C. A. McGonegal: "An Approach to the Approximation Problem for Nonrecursive Digital Filters", IEEE Trans. Audio Electroacoust., Vol. AU-18, pp 83-106, June 1970.

L. R. Rabiner and B. Gold: Theory and Application of Digital Signal Processing, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1975, pp 105-123.

c. Flow Chart: Page 9-107

d. Cross Reference Table: Page 9-222.

6. THEORY OF OPERATION

The basic concept of this filter design technique is as follows. A comb filter is used to synthesize N zeros around a circle of radius R in the Z-plane. The banks of parallel resonators are placed in cascade with the comb filter. The resonators are chosen such that once the desired passband region their poles exactly cancel the zeros. The weight applied to the resonators are samples of the desired frequency domain transfer function.

The block diagram of the digital filter simulated by the subroutine is shown in Figure DIGFSF-1. Since the phases at resonance of the consecutive :esonators differ by , a phase adjustment is inserted after each resonator. The location of the zeros synthesized by the comb filter are shown in Figure DIGFSF-2. The block diagram of a typical complex resonator is shown in Figure DIGFSF-3.

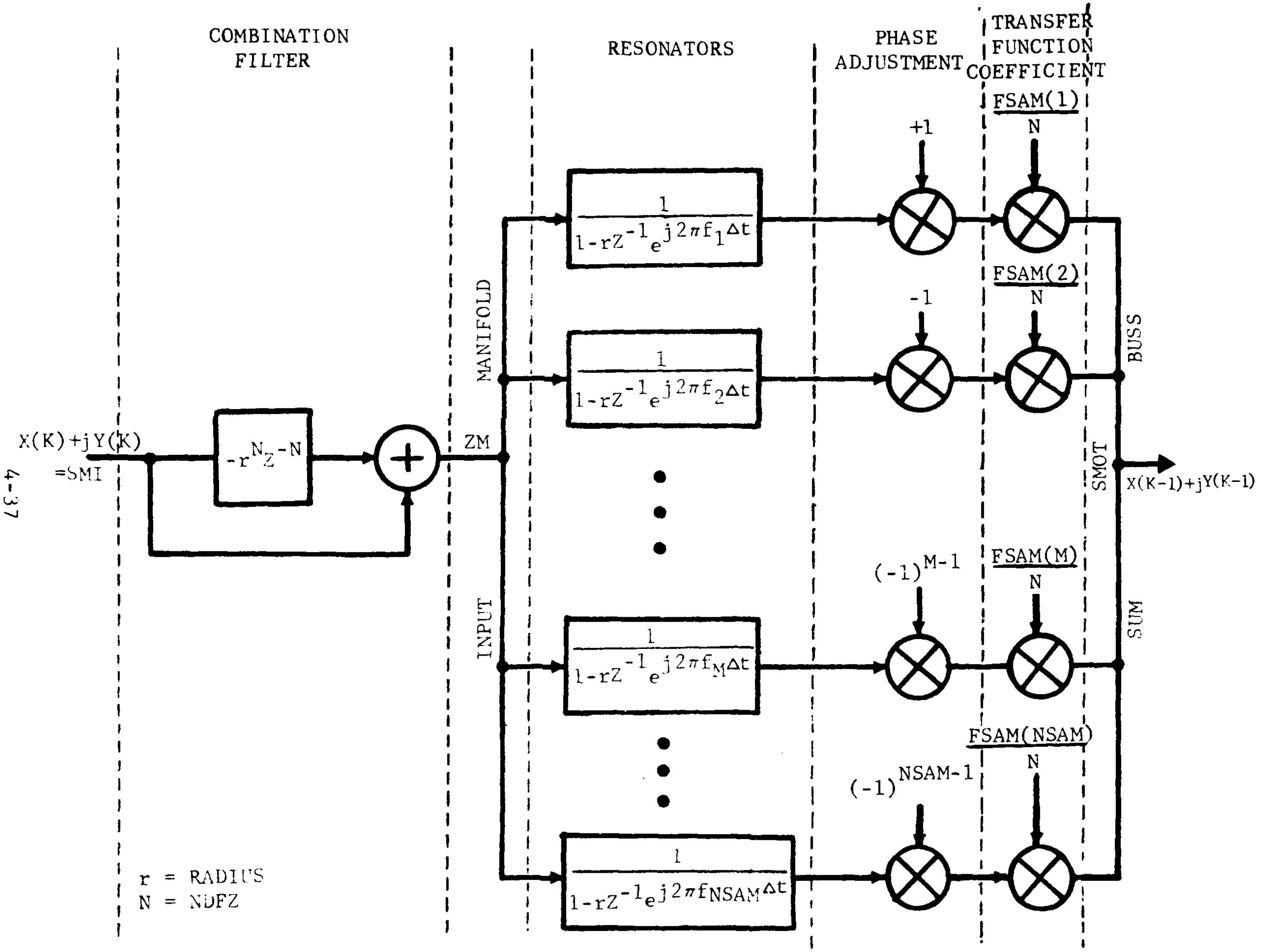


Figure DIGFSF-1 BLOCK DIAGRAM OF DIGFSF

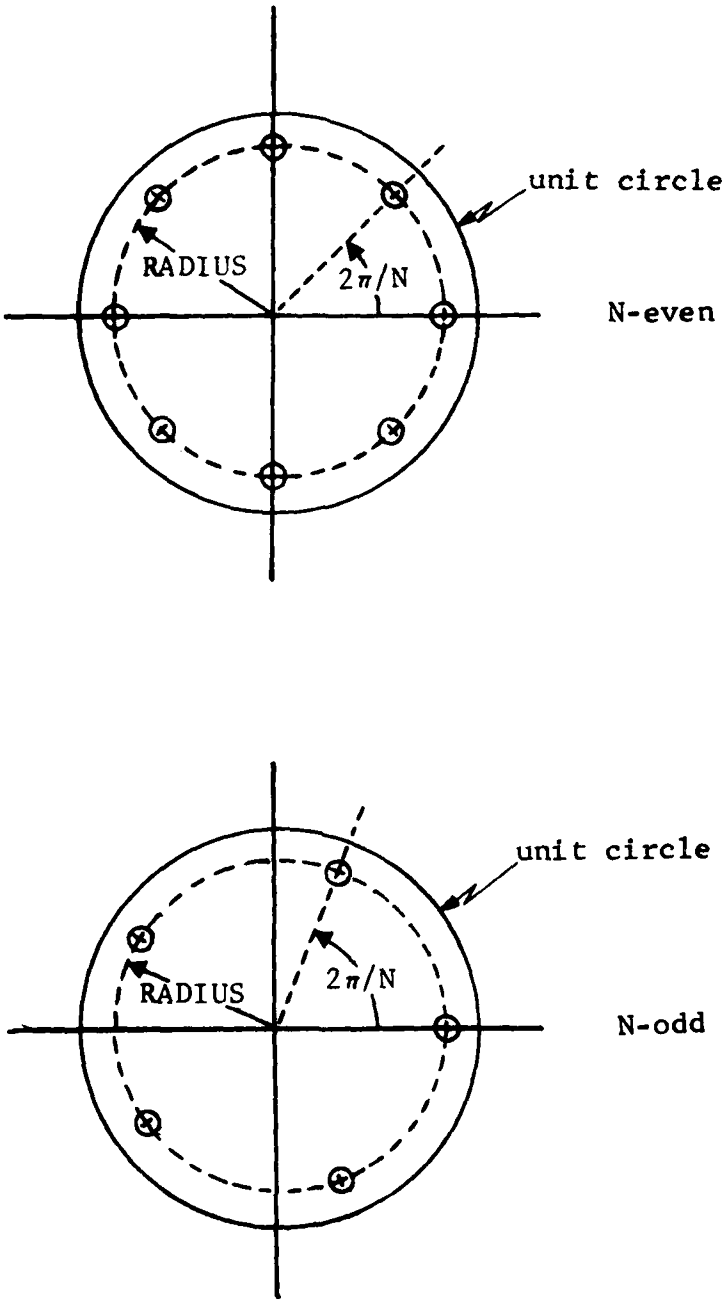
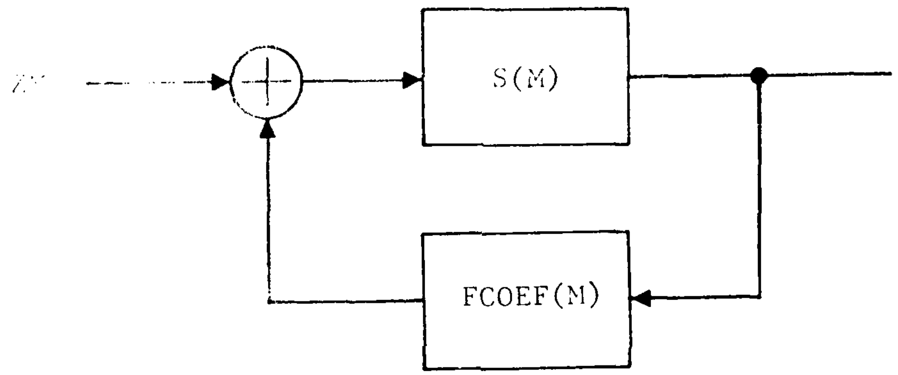


Figure DIGFSF-2 COMBINATION FILTER ZERO LOCATIONS IN THE Z-PLANE



where: $\mathrm{FCOEF(M)} = e^{j2\pi f_M \Delta t}$

$f_M = f_S\mathrm{(M\text{-}NDC)/NDFZ}$

Figure DIGFSF-3 BLOCK DIAGRAM OF A COMPLEX RESONATOR

```
      SUBROUTINE [illegible](X,Y)
      COMMON/BLK1/ VAR(200),FSAM(100)
      EQUIVALENCE (VAR(195), RADIUS ) , (VAR(196), NSAM ),
     *            (VAR(197), NOFZ    )
      COMPLEX FSAM,FF(200),FCOEF(100),E(100)
      COMPLEX SM,SMLT,EM,CZERO/(0.0,0.0)/
      DIMENSION X(1),Y(1)
      DATA [illegible]/,PI2/6.2831853/
      [illegible]=[illegible](X(195))
      GO TO 50
      ENTRY COMPSC(X,Y)
      [illegible]=1
   50 CONTINUE
      N[illegible]=NSAM/2+1
      ANG=[illegible]
      [illegible]=PI2*A[illegible]
      [illegible]=1.0
      DO 100 K=1,NSAM
      [illegible]
```

```
CARD NO     ****                       CONTENTS

                  FCOEF(K)=CMPLX(COS(PH),SIN(PH))*RADIUS
                  FSAM(K)=FSAM(K)*DELSC
                  DELSC=-DELSC
                  S(K)=CZERO
            100   CONTINUE
                  NDC=NDFZ+1
                  DO 105 K=1,NDC
                  FF(K)=CZERO
            105   CONTINUE
                  DELSC=RADIUS**NDFZ
                  IRP=1
                  ISP=1
                  ENTRY ECMFSF(X,Y)
                  DO 200 K=1,NPTS
                  IRP=IRP+1
                  SMI=CMPLX(X(K),Y(K))
                  FF(ISP)=SMI*DELSC
                  ZM=SMI-FF(IRP)
                  ISP=IRP
                  IF(IRP.EQ.NDC) IRP=0
                  SMOT=CZERO
                  DO 300 M=1,NSAM
                  SMOT=SMOT+S(M)*FSAM(M)
                  S(M)=S(M)*FCOEF(M)+ZM
            300   CONTINUE
                  X(K)=REAL(SMOT)*XNI
                  Y(K)=AIMAG(SMOT)*XNI
            200   CONTINUE
                  RETURN
```

SUBROUTINE DIGTFL

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
DIGTFL	NL-2	422,423

2. PURPOSE:

This subroutine is used to simulate a digital transversal filter.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NTAPS	R	I	Number of taps
ITAP	R	I	Array containing the tap specifications. The coefficient for the Kth tap are the following: ITAP(1,K) = numerator of tap gain ITAP(2,K) = denominator of tap gain ITAP(3,K) = delay in sampling increments

4. CALLING SEQUENCES:

CALL DIGTFL(IN,IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. This device is typically used as a digital phase decoder.

b. Flow Chart: Page 9-95

c. Cross Reference Table: Page 9-220

6. THEORY OF OPERATION

The block diagram of the digital transversal filter simulated by this module is shown in Figure DIGTFL-1. All calculations performed by this module are in integer format.

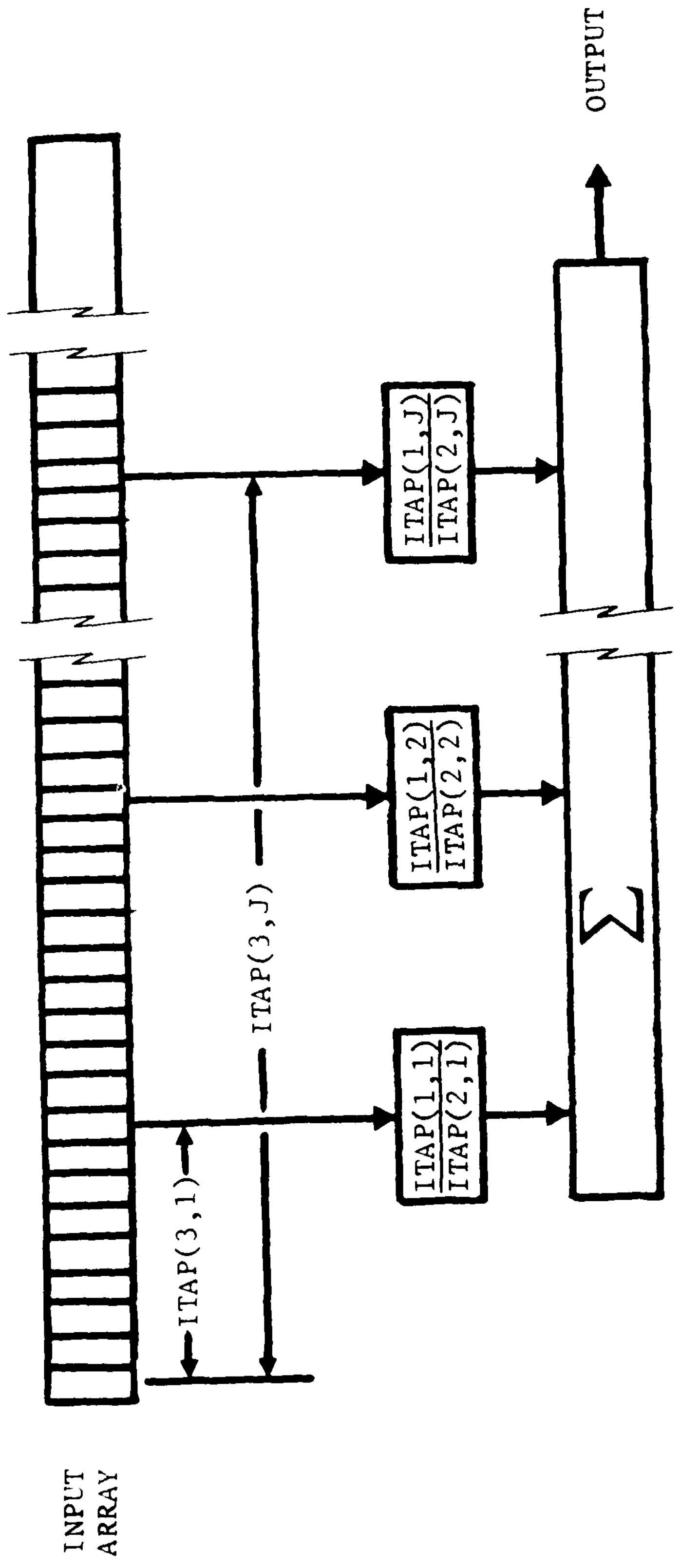


Figure DIGTFL-1 BLOCK DIAGRAM OF DIGTFL

```
 2542            SUBROUTINE DIGTFL(IN,IOUT)
 2543            COMMON/BLK1/BK1(200),ITAP(3,100)
 2544            DIMENSION IN(1),IOUT(1)
 2545            EQUIVALENCE  (BK1(101), NTAPS        )
 2546            DATA N193,N194,N195,N196/-3,-2,-1,0/
 2547            NSTOP=IN(N193)
 2548            NSTOP=NSTOP-ITAP(3,NTAPS)
 2549            DO 100 J=1,NSTOP
 2550            IA=0
 2551            DO 200 K=1,NTAPS
```

```
08/11/75       INPUT LISTING                    AUTOFLOW CHART SET - FWD/SCL  RADSIM

 CARD NO.      ****                          CONTENTS                                 ****

 2552            IA=IA+ (ITAP(1,K)* IN(J+ITAP(3,K)) ) / ITAP(2,K)
 2553        200 CONTINUE
 2554            IOUT(J)=IA
 2555        100 CONTINUE
 2556            IOUT(N193)=NSTOP
 2557            IOUT(N194)=IN(N194)
 2558            IOUT(N195)=IN(N195)
 2559            RETURN
 2560            END
```

SUBROUTINE ECM

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
ECM	SOU-1S	512

2. PURPOSE:

This subroutine simulates a noise modulation jammer.

3. INPUT PARAMETERS:

Name	O/R	T	Description
TIME	R	F	Elapsed time since beginning of simulation
JRNGØ	O	F	True jammer range corresponding to TIME = 0.0
JRSIM	R	F	Jammer range to be used in computing the starting time of the jammer.
JMAZ	R	F	Jammer azimuth angle
JHGT	O	F	Jammer height above ground
JERP	R	F	Jammer effective radiated power
JFMBW	O	F	Jammer swept bandwidth
JPW	O	F	Jammer on time for pulsed jammers
JFØ	O	F	Jammer center frequency offset from radar center frequency
JVEL	O	F	Jammer radial velocity with respect to the radar site
JPERΘD	O	F	Jammer pulse repetition interval for pulsed jammer

Name	O/R	T	Description
N2	R	I	2**N2 = total number of signal storage array elements available
NDFZ	R	I	Number of zeros to be synthesized by the frequency sampling digital filter (DIGFSF).

In addition to the above parameters the characteristics of the jammer spectrum must be specified via the parameters of DIGFSF. These parameters are also members of NLS12.

4. CALLING SEQUENCES:

CALL ECM (X, Y)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The digital filter routine DIGFSF is used to specify the spectral characteristics of the jammer.

b. Flow Chart: Page 9-104

c. Cross Reference Table: Page 9-221

6. THEORY OF OPERATION

This module has four modes of operation:

(1) Spot noise/Barrage noise

$JPW = 0.0$

(2) Swept noise (single sweep)

$JFMBW \neq 0.0$
$JPW \neq 0.0$
$JPEROD = 0.0$

(3) Swept noise (repeating sweep)

$JFMBW \neq 0.0$
$JPW \neq 0.0$
$JPEROD = JPW$

(4) Pulsed noise

$JPW \neq 0.0$
$JPEROD > JPW$

Dependeing on the range and azimuth positioning of the jammer with respect to the target either a self screening or a standoff jammer can be simulated.

A block diagram of this module is shown in figure ECM-1. The output of the ECM simulation is summed with the data in arrays X and Y.

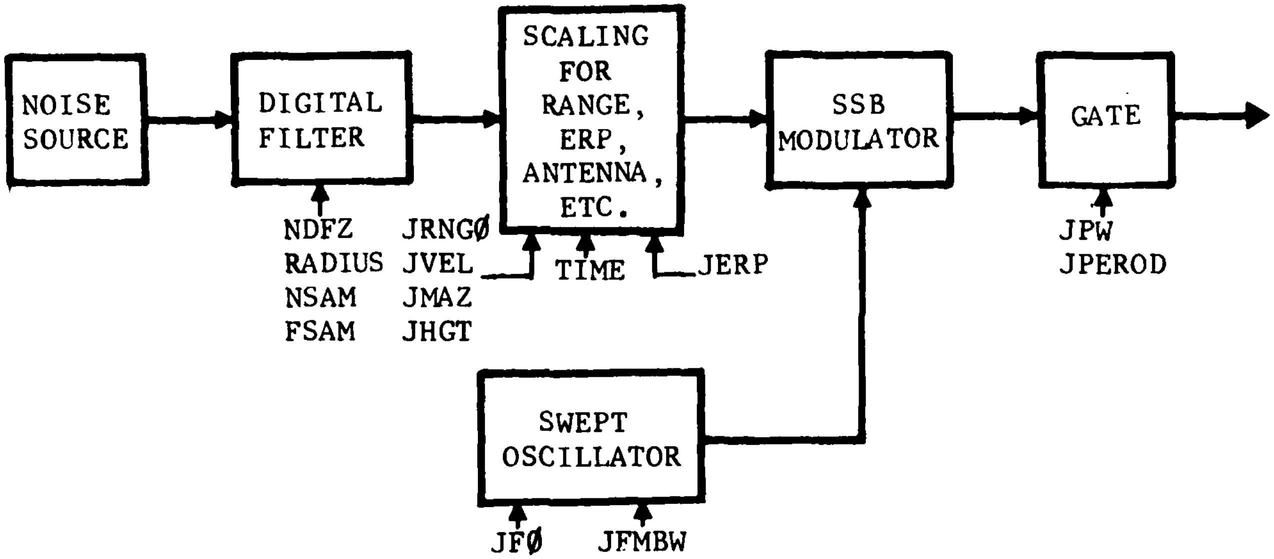


Figure ECM-1 BLOCK DIAGRAM OF ECM

```
CARD NO      ****                    CONTENTS                    ****

[illegible]         SUBROUTINE ECM(X,Y)
[illegible]         COMMON/[illegible]X/ XK(1500)
[illegible]         COMMON/BLKKNO/ DMY(E),YR
[illegible]         EQUIVALENCE (XK( 16), TIME  ) , (XK(171), JRNGO ),
[illegible]        *        (XK(172), JESIM ) , (XK(173), JMAZ  ),
[illegible]        *        (XK(174), JHOT  ) , (XK(175), JEFF  ),
[illegible]        *        (XK(176), JFMEW ) , (XK(177), JPW   ),
[illegible]        *        (XK(178), JFO   ) ,
[illegible]        *        (XK([illegible]), JVEL  ) , (XK([illegible]), JFREQ)
[illegible]        *       ,(XK( [illegible]), N2    ) , (XK([illegible]), NOR   ),
[illegible]         DIMENSION X(1),Y(1)
[illegible]         DATA N1,N2,N3/[illegible]/
[illegible]         REAL JRNG,JESIM,JMAZ,JHOT,JEFF,JFMEW,JPW,JFO
[illegible]         REAL JVEL,JFREQ,JSTART,JRNGO
[illegible]         NE1=[illegible](AINT(N3))
[illegible]         DELT=[illegible]
[illegible]         JSTART=(JESIM+JVEL*TIME)/[illegible]
[illegible]         [illegible]
[illegible]         JRNG=JRNGO+TIME*JVEL
[illegible]         IF(JRNG.LE.0.0) GO TO 5
[illegible]         [illegible](EXP[illegible](JHOT/JRNG)*57.[illegible])
[illegible]         [illegible](AINT(JMAZ) * [illegible])
[illegible]         [illegible]
[illegible]         [illegible]
[illegible]         CALL [illegible](XR,YR)
[illegible]         DO 100 J=1,NE1
[illegible]         [illegible]=RANF([illegible])
[illegible]         CALL [illegible](XR,YR)
[illegible]     100 CONTINUE
[illegible]         [illegible]
[illegible]         [illegible]
[illegible]         IF(JPW.NE.0.0) GO TO 200
[illegible]         [illegible]
[illegible]         [illegible]
[illegible]         [illegible]
[illegible]         [illegible]
[illegible]         GO TO [illegible]
```

This model is incomplete since the TWT does not produce saturated noise and the output of this model is not properly scaled for range effects. This will be accomplished in the near future.

```
  CARD NO         ****                              CONTENTS                                ****

   2784          200   JBG=IFIX(JSTART/DELT)
   2785                 IF(JBG.LT.1) JBG=1
   2786                 JST=IFIX(JPW/DELT)+JBG
   2787                 IF(JST.GT.NTTL) JST=NTTL
   2788                 FSTRT=JF0-JFMBW*0.5
   2789                 CHIRP2=JFMBW/JPW*0.5
   2790                 IF(JPERDD.EQ.0.0) GOTO 300
   2791                 NFPT=1
   2792                 IF(JPERDD.GT.JPW) GOTO 290
   2793                 JCTOFF= 1
   2794                 GO TO   300
   2795          290   JCTOFF=IFIX((JPERDD-JPW)/DELT)+1
   2796          300   IF(NFTS.GE.NTTL) GOTO 310
   2797                 K=NFPTS+1
   2798                 DO 301 J=K,NTTL
   2799                 X(J)=0.0
   2800                 Y(J)=0.0
   2801          301   CONTINUE
   2802                 X(N192)=DELT(NTTL)
   2803                 Y(N193)=X(N193)
   2804          310   T=0.0
   2805                 DO 500 J=JBG,JST
   2806                 XF=FRAND(0)
   2807                 CALL COMPGF(XF,YF)
   2808                 PH= (FSTRT+CHIRP2*T)*T
   2809                 FF= AMOD(PH,1.0)*PI2
   2810                 C=COS(FF)
   2811                 S=SIN(FF)
   2812                 X(J)=X(J)+C*XF-S*YF
   2813                 Y(J)=Y(J)+C*YF+S*XF
   2814                 T=T+DELT
   2815          500   CONTINUE
   2816                 IF(NFPT.EQ.0) RETURN
   2817                 JBG=JST+JCTOFF
   2818                 IF(JBG.GT.NTTL) RETURN
   2819                 JST=JBG+IFIX(JPW/DELT)
   2820                 IF(JST.GT.NTTL) JST=NTTL
   2821                 GO TO 310
   2822                 END
```

SUBROUTINE FGENXY

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
FGENXY	SOU-1 or LTI-3	420
FGENMP	SOU-1 or LTI-3	421

2. PURPOSE:

This subroutine is used to simulate the radar waveform generator. In general, the waveform generator subsystem in a radar drives the power amplifier.

3. INPUT PARAMETERS

a. MODE #1; single pulse, internal modulation

Name	O/R	T	Description
FS	R	F	Simulation sampling rate
FØ	R	F	Center frequency of output waveform
INORM	R	I	Normalization flag
TI	R	F	Time increment between output waveform samples
NPWTX	R	I	Set = 0 for this mode
FMBW	0	F	Linear FM bandwidth
NSUBP	R	I	Set = 0 for this mode
RISTIM	0	F	Rise Time
FALTIM	0	F	Fall Time
TSTART	0	F	Start time of the output waveform
PW	R	F	Pulsewidth

Name	O/R	T	Description
VPEAK	R	F	Peak output voltage
N2	R	I	Simulation parameter used to specify maximum array length

Figure FGENXY-1(a) shows the relationship between some of the input parameters and the output waveform.

b. MODE #2; single pulse, User modulation function

Name	O/R	T	Description
FS	R	F	Simulation sampling rate
FØ	0	F	Center frequency of output waveform
INORM	R	I	Normalization flag
TI	R	F	Time increment between output waveform samples
NPWTX	R	I	Number of points in user specified modulation function
WT	R	F	Array containing the user specified modulation function. The specification for the Jth sample of the weighting function is the following: WT(1,J) = Gain WT(2,J) = Time WT(3,J) = Phase angle
NSUBP	R	I	Set = 0 for this mode
FSTART	0	F	Starting frequency at time = TSTART
CHIRP	0	F	Linear FM sweep rate
TSTART	0	F	Start time of the output waveform

Name	O/R	T	Description
VPEAK	R	F	Peak output voltage
N2	R	I	Simulation parameter used to specify maximum array length

NOTE: Either FO or FSTART & FMBW must be specified.

Figure FGENXY-1(b) shows the relationship between some of the input parameters and the output waveform.

c. MODE #3; phase coded waveform

Name	O/R	T	Description
FS	R	F	Simulation sampling rate
FØ	0	F	Center frequency of output waveform
INORM	R	I	Normalization flag
TI	R	F	Time increment between output waveform samples
NPWTX	R	I	Set = 0 for this mode
NSUBP	R	I	Number of subpulses
SPW			Subpulse width
PCODE	R	F	Array containing the phase code
SWTIM	0	F	Switching time between subpulses
RISTIM	0	F	Rise time
FALTIM	0	F	Fall time
FSTART	0	F	Starting frequency at time = TSTART
CHIRP	0	F	Linear FM sweep rate
TSTART	0	F	Start time of the output waveform

Time	O/R	T	Description
VPEAK	R	F	Peak output waveform
N2	R	I	Simulation parameter used to specify maximum array length

NOTE: Either FO or FSTART & FMBW must be specified.

Figure FGENXY-1(c) shows the relationship between some of the input parameters and the output waveform.

4. CALLING SEQUENCES:

CALL FGENXY (X,Y)

Where: X contains the Output Waveform - R
Y contains the Output Waveform - I

CALL FGENMP (X,Y)

Where: X contains the Output Waveform - M
Y contains the Output Waveform - P

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The number of waveform samples to be generated must be less than 2**N2 where N2 ≤ 13.

b. The normalization flag, INORM, is used to indicate whether the output is to be interpreted as a matched filter impulse response or as a frequency generator output waveform.

If INORM = 0, 2 or 4 the output represents a frequency generator output.
If INORM = 1 or 3 the output represents the impulse response of an ideal matched filter and therefore TSTART is set equal to zero and no trailing zeros are added to the output. This procedure is required because the normalization procedure used in the Fourier transform routine is different for waveforms and impulse responses.

c. For Mode #3 the sign of NSUBP indicates whether a polyphase or binary phase code is to be generated.

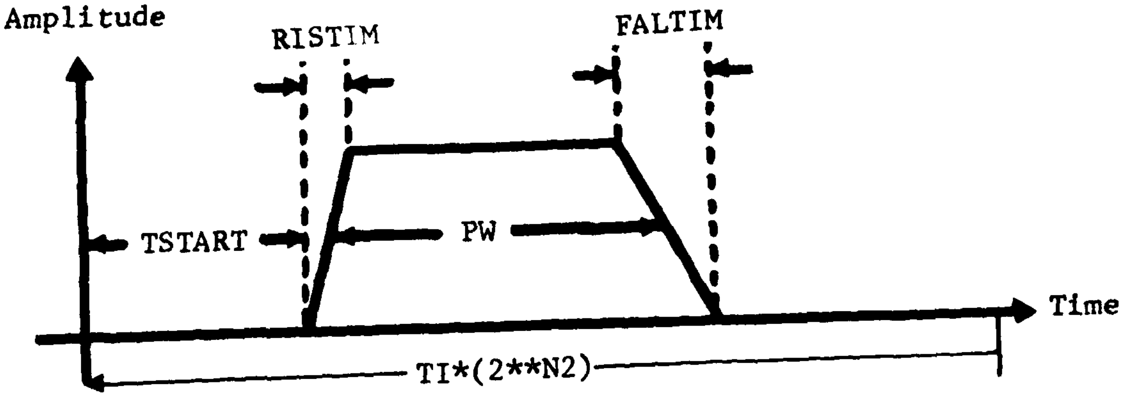


Figure FGENXY-1a OUTPUT WAVEFORM ENVELOPE - MODE 1

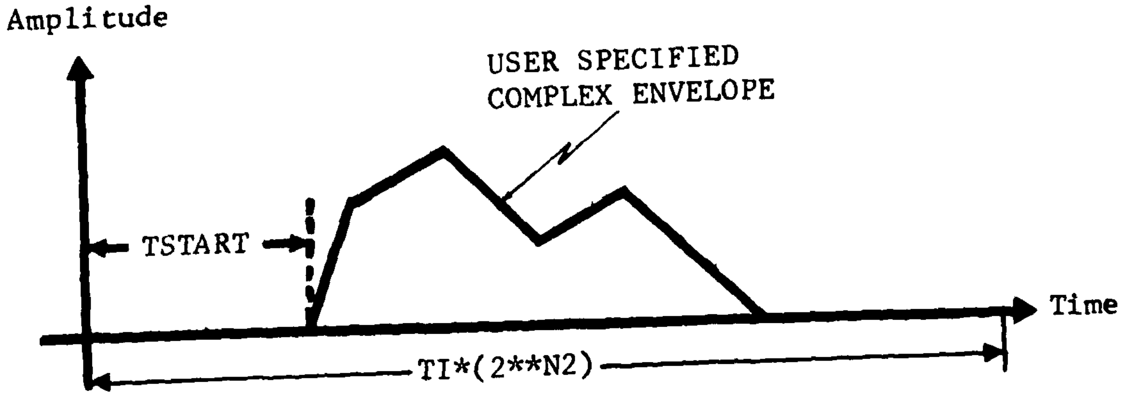


Figure FGENXY-1b OUTPUT WAVEFORM ENVELOPE - MODE 2

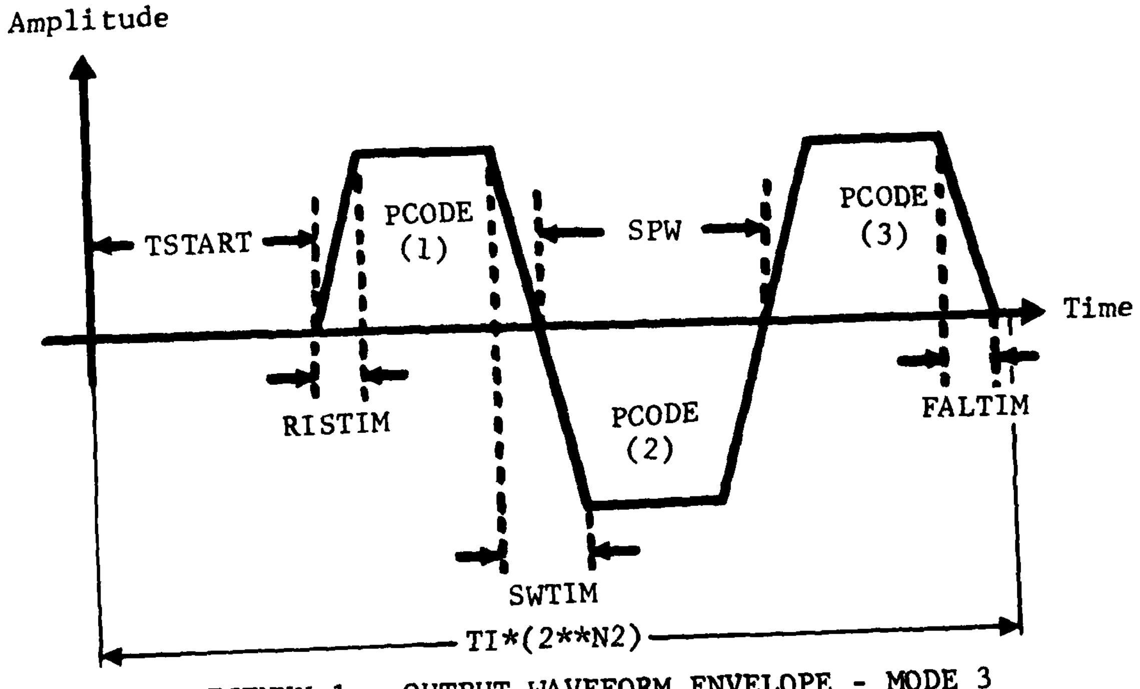


Figure FGENXY-1c OUTPUT WAVEFORM ENVELOPE - MODE 3

If NSUBP is positive the phase is either 0^{o} or 180^{o} with respect to the carrier. If NSUBP is negative the phase between subpulses is determined by linear interpolation.

d. Operation of this module in Mode #3 causes the phase code array (PCODE) to be copies into the storage array XMITPC for later use by the phase decoder routine, PHDEC. In this way the phase decoder is automatically slaved to the phase code of the waveform generator.

e. Abort error codes

1, User attempted to specify both a phase code and a complex envelope

2, User attempted to specify a complex envelope with less than 4 points

3, The time span of the waveform to be generated is greater than the simulation time span, i.e. TI*(2**N2).

104, An error occurred during execution of the weighting function routine, WEITCP or WEITMP.

f. Flow Chart: Page 9-72

g. Cross Reference Table: Page 9-217.

6. THEORY OF OPERATION

This simulation is used to simulate the master frequency source of a radar system. The basic mechanization equation is given by the following expression:

$$S(t) = a(t) \exp \; j(2\pi f_0 t + \theta(t))$$

The envelope a(t) and the phase modulation term θ(t) are determined either directly by the user (MODE #2) or indirectly through parameters (MODE #1 or #2).

This module can also handle a staggered PRF by specifying NPRIS and PRI(J) in the system module (301) namelist. NPRIS is the number of pulse repetitions and must be ≤ 11. PRI(J) are the intervals in seconds of the pulses. For example, under $NL301 might be included "..., NPRIS = 4, PRI = 1.Ø, Ø.9, 1.1, 1.Ø$."

```
      SUBROUTINE FGENXY(X,Y)
      COMMON/BLK1/ VAR(200), WT(3,100)
      COMMON/PHCODE/ XMITPC(302)
      DIMENSION PCODE(300),X(1),Y(1)
      EQUIVALENCE (WT(1,1) ,PCODE(1))
      EQUIVALENCE (VAR(  2) ,FS     ) , (VAR(  8) , FC     ),
     *            (VAR(  9), INCRM    ),
     *          (VAR( 12) ,TI     ) , (VAR( 37) , NPWT  ),
     *          (VAR(38) , ORIG   ) , (VAR( 42) , CHIRP ) ,
     *          (VAR( 43) , FMBW  ) , (VAR( 44) ,NPWTX ),
     *          (VAR( 45) , SPW    ) , (VAR( 46) , NSUBP ),
     *          (VAR( 97) , SWTIM ) , (VAR( 98) , RISTIM),
     *          (VAR( 99) , FALTIM) , (VAR(100) , TSTART),
     *          (VAR( 41) , FSTART) , (VAR( 40) , PW     ),
```

H-57a

```
 CARD NO        ****                        CONTENTS                                        ****

  2088               *           (VAR(129) , VPEAK) , (VAR(  1) , N2     )
  2089                DATA N193,N194,N195,DTR/-3,-2,-1,1.7453292E-02/
  2090               DATA XCIRC/2.7777E-03/,PI2/6.2831853/
  2091                ITYPE=1
  2092                GOTO 100
  2093                ENTRY FGENMF(X,Y)
  2094                ITYPE=2
  2095          100   IFMM=0
  2096                IF(NSUBF.GE.0) GOTO 110
  2097                NSUBF=-NSUBF
  2098                IFMM=1
  2099          110   IF(CHINP.EQ.0.0) FSTART=FC
  2100                NTTL=2**N2
  2101                IF(NSUBF.EQ.0) GOTO 200
  2102                IF(NPWTX.NE.0) CALL ABORT(1)
  2103                SUBFP=PCODE(1)
  2104                XMITFC(301)=BCDL(NSUBF)
  2105                XMITFC(302)=SPW
  2106                DO 500 J=1,NSUBF
  2107                XMITFC(J)=FCODE(J)
  2108          500   CONTINUE
  2109                GO TO 600
  2110          200   CONTINUE
  2111                IF(NPWTX.EQ.0) GOTO 300
  2112                IF(NPWTX.LT.4) CALL ABORT(2)
  2113                SPW=WT(2,NPWTX)
  2114                FISTIM=T1
  2115                FALTIM=T1
  2116                GOTO 400
```

```
      SPW=PW+(RISTIM+FALTIM)*0.5
      IF(CHIRP.EQ.0.0) CHIRP=FMBW/PW
      FSTART=FC-CHIRP*(PW+RISTIM)*0.5
  400 CONTINUE
      SUBPH=0.0
      NSUBP=1
      SWTIM=RISTIM+FALTIM
  600 NSTRT=0
      IF(INORM.EQ.1.OR.INORM.EQ.3) GOTO 651
      NSTRT=IFIX(TSTART*FS)
      TSTART=T1*FLOAT(NSTRT)
      NSTRT=NSTRT+1
      DO 650 J=1,NSTRT
      X(J)=0.0
      Y(J)=0.0
  650 CONTINUE
  651 NS=IFIX( FLOAT(NSUBP)*SPW*FS )+NSTRT
      IF(NS.GT.NTTL) CALL ABORT(3)
      TIME=0.0
      K=1
      J=NSTRT
*
*****    LEADING EDGE OF XMITTED WAVEFORM    *****
*
      IREG=0
      ASLOPE=VPEAK/RISTIM
      IF(IPHM.EQ.0) ASLOPE=ASLOPE*COS(SUBPH*DTR)
      PSLOPE=0.0
```

```
 CARD NO        ****                    CONTENTS                    H-58a  ****~

   2146                  TORIG=0.0
   2147                  PORIG=SOLPH
   2148                  AORIG=0.0
   2149                  BREAK=RISTIM
   2150          700     J=J+1
   2151                  TIME=TIME+TI
   2152          800     IF(TIME.GE.BREAK) GOTO 900
   2153                  DTIM=TIME-TORIG
   2154                  PH=(FSTART+CHIRP*0.5*TIME) * TIME
   2155                  IF(IPHM.EQ.1) PH=PH+(PORIG+PSLOPE*DTIM)*XCIFC
   2156                  AMPL=AORIG + ASLOPE*DTIM
   2157          860     IF(ITYPE.EQ.1) GOTO 870
   2158                  X(J)=AMPL
   2159                  Y(J)=PH*360.0
   2160                  GOTO 700
   2161          870     PHAS=(PH-FLOAT(IFIX(PH)))*PI2
   2162                  X(J)=AMPL*COS(PHAS)
   2163                  Y(J)=AMPL*SIN(PHAS)
   2164                  GOTO 700
   2165          900     IREG=IREG+1
   2166                  TORIG=BREAK
   2167                  IF(IREG.EQ.1) GOTO 910
   2168                  IF(IREG.EQ.4) GOTO 1000
   2169                  IF(IREG.NE.3) GOTO 920
   2170                  IREG=IREG-2
   2171         *
   2172         *****       TOP OF SUBPULSE        *****
   2173         *
   2174          910     BREAK=BREAK+SPW-SWTIM
```

```
            PSLOPE=0.0
            PORIG=SUBPH
            ASLOPE=0.0
            AORIG=VPEAK
            IF(IPHM.EQ.0) AORIG=VPEAK*COS(SUBPH*DTR)
            GOTO 800
      *
      *****     SWITCH REGION        *****
      *
       920  K=K+1
            IF(K.GT.NSUBP) GOTO 950
            TPEAK=TPEAK+SWTIM
            SUBPH=PCODE(K)
            PSLOPE= (SUBPH - PORIG)/SWTIM
            IF(IPHM.EQ.0) ASLOPE=(VPEAK*COS(SUBPH*DTR)-AORIG)/SWTIM
            GOTO 800
      *
      *****     TRAILING EDGE OF XMITTED WAVEFORM    *****
      *
       950  TPEAK=TPEAK+FALTIM
            ASLOPE=-AORIG/FALTIM
            IREG=3
            GOTO 800
      1000  IF(INORM.EQ.1.OR.INORM.EQ.3) GOTO 1501
            DO 1500 K=1,NTTL
            X(K)=0.0
            Y(K)=0.0
      1500  CONTINUE
            GOTO 1502
```

```
  CARD NO         ****                        CONTENTS                                  ****

   2204          1501  NTTL=J-1
   2205          1502  X(N193)=SNGL(NTTL)
   2206                X(N194)=0.0
   2207                X(N195)=TI
   2208                Y(N193)=X(N193)
   2209                Y(N194)=X(N194)
   2210                Y(N195)=X(N195)
   2211                IF(NEWTX.EQ.0) RETURN
   2212                CFLG=ISTART
   2213                NEWT=NEWTX
   2214                IF(ITYPE.NE.1) GOTO 1999
   2215                DO 2000 J=1,NEWTX
   2216                XM=WT(1,J)
   2217                WT(1,J)=XM*COS(WT(3,J)*DTR)
   2218                WT(2,J)=XM*SIN(WT(3,J)*DTR)
   2219          2000  CONTINUE
   2220                CALL WRITCF(X,Y,11,00)
   2221                RETURN
   2222          1999  CALL WRITRF(X,Y,11,00)
   2223                RETURN
   2224          1500  CALL ABORT(104)
   2225                RETURN
   2226                END
```

SUBROUTINE FILT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
FILT	LTI-1	407,408

2. PURPOSE:

This subroutine simulates a continuous filter which is defined by an S-domain polynomial transfer function.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NZ	R	I	Number of zeros in the transfer function.
NP	R	I	Number of poles in the transfer function.
SF	R	F	Scale factor.
FZERO	O	F	Array containing the zero specifications. The s-plane location for the Kth zero is given by the following coefficients: FZERO(1,K) = real component FZERO(2,K) = imaginary component
FPOLE	O	F	Array containing the pole specifications. The s-plane location for the Kth pole is given by the following coefficients: FPOLE(1,K) = real component FPOLE(2,K) = imaginary component

4. CALLING SEQUENCES:

CALL FILT(X,Y)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. A purely imaginary pole may not be specified within the frequency extent of the input waveform. This would cause either zero or an extremely small number to occur in the denominator of the transfer function.

b. Flow Chart: Page 9-55

c. Cross Reference Table: Page 9-215

6. THEORY OF OPERATION

The S-domain transfer function for a general filter can be represented by the ratio of two polynomials as shown in the following expression

$$H(S) = \frac{(S-Z_1)(S-Z_2)(S-Z_3).\,.\,.(S-Z_{NZ})}{(S-P_1)(S-P_2)(S-P_3).\,.\,.(S-P_{NP})}\cdot SF$$

In general, H(S) is a complex function of the frequency variable s. The filter output signal $S_o(f)$ can be determined using the following expression

$$S_o(f) = H(f)\ S_i(F)$$

where $S_i(f)$ is the input signal representation in the frequency domain. The discrete representation of these equations used in the module are the following:

$$H(J) = SF \cdot \frac{\Pi_{K=1}^{K=NZ}\left\{ j\,\mathrm{FREQ}(J) - \left[\text{FZERØ}(1,K) + j\,\mathrm{FZERO}(2,K)\right]\right\}}{\Pi_{K=1}^{K=NP}\left\{ j\,\mathrm{FREQ}(J) - \left[\text{FPØLE}(1,K) + j\,\text{FPØLE}(2,K)\right]\right\}}$$

$$X(J)+j\ Y(J) = H(J) * \left[X(J)+j\ Y(J)\right]$$

where: FREQ(J) is the frequency associated with the Jth sample

```
      SUBROUTINE FILT( X, Y)
      COMMON/BLK1/ BK1(200),FZERO(2,50),FPOLE(2,50)
      EQUIVALENCE  ( NZ   ,BK1(72)) , ( NP   ,BK1(73)), ( SF   ,BK1(74))
      DIMENSION X(1), Y(1)
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      FREQ=X(N194)
      N = IBCOL(X(N193))
      DELF = X(N195)
      IF( SF .EQ. 0. ) SF = 1.0
      DO 100 J=1,N
      IF(NZ.EQ.0) GO TO 150
      DO 200 K=1,NZ
      WD = FREQ - FZERO(2,K)
      A = X(J)
```

```
          X(J) = -A *FZERO(1,K) - Y(J) * WD
          Y(J) =  A * WD - Y(J) * FZERO(1,K)
      200 CONTINUE
      150 IF(NP.EQ.0) GO TO 300
          DO 250 K=1,NP
          WD = FREQ -FPOLE(2,K)
          A = X(J)
          AA = FPOLE(1,K) * FPOLE(1,K) + WD * WD
          X(J) =(-A * FPOLE(1,K) + Y(J) * WD) / AA
          Y(J) =(-A * WD - Y(J) * FPOLE(1,K)) / AA
      250 CONTINUE
      300 X(J) = X(J) * SF
          Y(J) = Y(J) * SF
          FREQ = FREQ + DELF
      100 CONTINUE
          RETURN
          END
```

SUBROUTINE HET

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
HET	LTV-4	-

2. PURPOSE:

This subroutine simulates a single sideband modulator which is used to heterodyne waveforms.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FSHIFT	R	F	Frequency shift to be applied to the input waveform.

4. CALLING SEQUENCE:

CALL HET(X,Y)

where: X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA:

a. Flow Chart: Page 9-103

b. Cross Reference Table: Page 9-221

6. THEORY OF OPERATION:

The basic mechanization equation for this module is the following:

$$s'(t) = s(t) \times e^{j\,\mathrm{FSHIFT}\cdot t}$$

or

$$X'(t) + jY'(t) = \left[X(t)+jY(t)\right] \cdot \left[\cos\,(\mathrm{FSHIFT}\cdot t)+ j\,\sin(\mathrm{FSHIFT}\cdot t)\right]$$

The block diagram of a SSB modulator is shown in Figure HET-1.

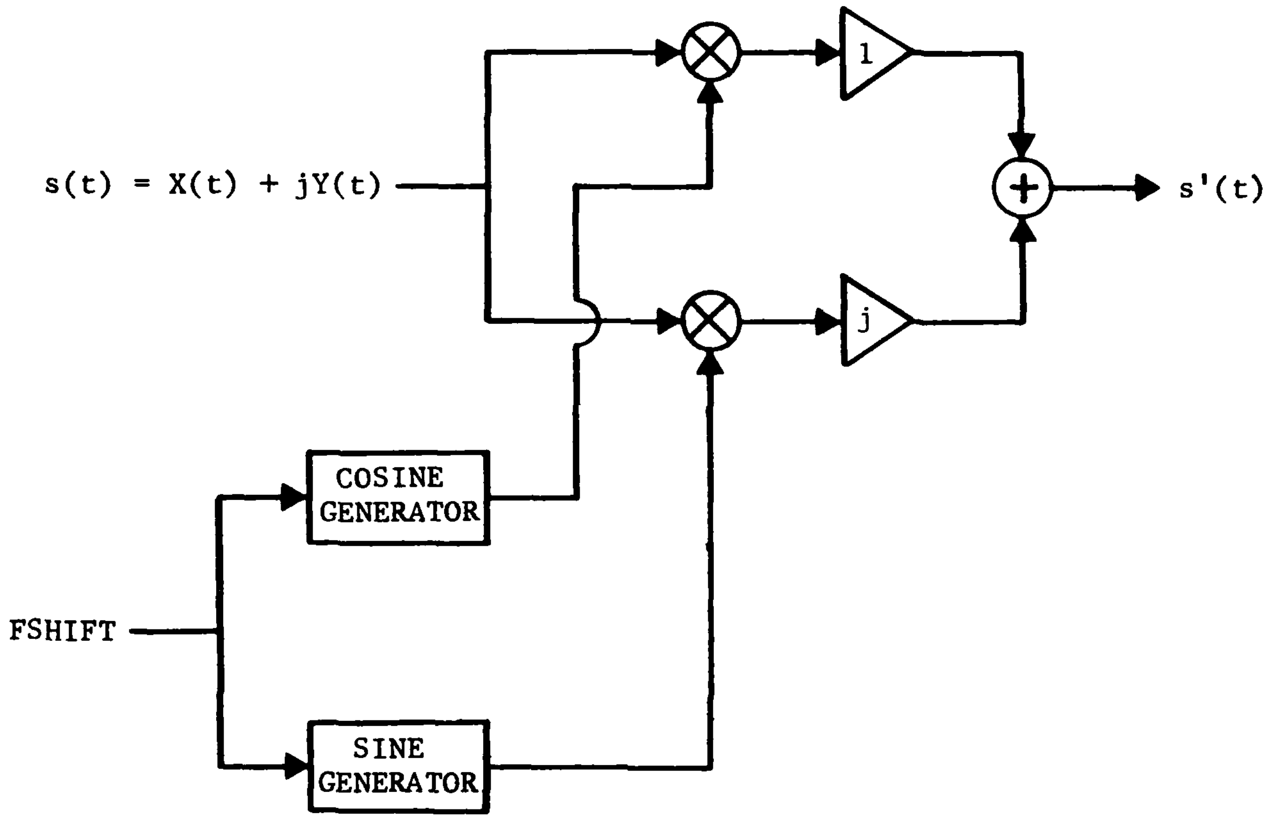


Figure HET-1 BLOCK DIAGRAM OF SSB MODULATOR

```
      SUBROUTINE HET(X,Y)
      DIMENSION X(1),Y(1)
      COMMON /PERI/ VAR(500)
      EQUIVALENCE (VAR( 15), FSHIFT )
      DATA N193,N194,N195,N196 /-3,-2,-1, 0/,PI2/6.283185/
      NETS=IFIX(X(N193))
      OR = X(N194)
      DEL=X(N195)
      DO100J=1,NETS
      THETA= OR*FSHIFT
      THETA= (THETA- FLOAT(IFIX(THETA)))*PI2
      C= COS(THETA)
      S= SIN(THETA)
      TMP=X(J)
      X(J)= X(J)*C - Y(J)*S
      Y(J)= TMP*S + Y(J)*C
      OR= OR +DEL
  100 CONTINUE
      RETURN
      END
```

SUBROUTINE HWDET

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
HWDET	NL-1	414,415
FWDET	NL-1	416,417
SQDET	NL-1	418,419
HLIM	NL-1	438,439
IHWDET	NL-1	445,446
IFWDET	NL-1	447,448
ISQDET	NL-1	449,450
IHLIM	NL-1	442,443

2. PURPOSE:

This subroutine simulates a signal detector. Depending on the entry point, a half-wave detector, a full-wave detector, a square-law detector, or a hard limiter is simulated.

3. INPUT PARAMETERS:

NONE

4. CALLING SEQUENCES AND THEORY OF OPERATION

a. Half-Wave Detector

CALL HWDET(XIN,XOUT)

Where: XIN contains the Input Waveform

XOUT contains the Output Waveform

$$\text{XOUT}(J) = \begin{cases} \text{XIN}(J) & ; \text{ if XIN}(J) > 0.0 \\ 0.0 & ; \text{ otherwise} \end{cases}$$

b. Full-Wave Detector

CALL FWDET(XIN,XOUT)

Where: XIN contains the Input Waveform

XOUT contains the Output Waveform

$XOUT(J) = |XIN(J)|$

c. Square-Law Detector

CALL SQDET(XIN,XOUT)

Where: XIN contains the Input Waveform

XOUT contains the Output Waveform

$XOUT(J) = XIN(J)*XIN(J)$

d. Hard Limiter

CALL HLIM(XIN,XOUT)

Where: XIN contains the Input Waveform

XOUT contains the Output Waveform

$$\begin{aligned} XOUT(J) &= 1.0 \ ; \text{ if } XIN(J) \geq 0.0 \\ &= -1.0; \text{ if } XIN(J) < 0.0 \end{aligned}$$

e. Digital Half-Wave Detector

CALL IHWDET(IN,IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

$$\begin{aligned} IOUT(J) &= IOUT(J) \ ; \text{ if } IOUT(J) > 0 \\ &= 0 \qquad ; \text{ otherwise} \end{aligned}$$

f. Digital Full-Wave Detector

CALL IFWDET(IN,IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

IOUT(J) = IABS(IN(J))

g. Digital Square-Law Detector

CALL ISQDET(IN,IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

IOUT(J) = IN(J)*IN(J)

h. Digital Hard Limiter

CALL IHLIM(IN,IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

```
IOUT(J) = 1 ; IN(J) > 0
        = 0 ; IN(J) = 0
        = 1 ; IN(J) < 0
```

5. <u>RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA</u>

a. Flow Chart: Page 9-137

b. Cross Reference Table: Page 9-225

```
      SUBROUTINE HWDET( XIN, XOUT)
      DIMENSION  XIN(1),XOUT(1),IN(1),IOUT(1)
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      NS = IBOOL(XIN(N193))
      DO 20 I=1,NS
      XOUT(I) = XIN(I)
   20 IF( XOUT(I) .LT. 0.0 ) XOUT(I) = 0.0
      GO TO 100
C
      ENTRY FWDET( XIN, XOUT)
      NS = IBOOL(XIN(N193))
      DO 40 I=1,NS
   40 XOUT(I) = ABS(XIN(I))
      GO TO 100
C
      ENTRY SWDET( XIN, XOUT)
      NS = IBOOL(XIN(N193))
      DO 80 I=1,NS
   80 XOUT(I) = XIN(I)*XIN(I)
      GO TO 100
C
      ENTRY HEIM(XIN,XOUT)
      NS=IBOOL(XIN(N193))
      DO 60 I=1,NS
      IF(XIN(I)) 61,62,62
   61 XOUT(I)=-1.0
      GO TO 60
   62 XOUT(I)=1.0
   60 CONTINUE
C
  100 XOUT(N193) = XIN(N193)
      XOUT(N194) = XIN(N194)
      XOUT(N195) = XIN(N195)
      RETURN
C
      ENTRY IFWDET(IN,IOUT)
```

```
 CARD NO        ****                              CONTENTS                              ****

  3596              NS= IN(N195)
  3597              DO 30 I=1,NS
  3598              IF(IN(I)) 31,31,32
  3599           31 IOUT(I)=0
  3600              GO TO 30
  3601           32 IOUT(I)=IN(I)
  3602           30 CONTINUE
  3603              GO TO 200
  3604        C
  3605              ENTRY IFMDET(IN,IOUT)
  3606              NS= IN(N195)
  3607              DO 60 I=1,NS
  3608           60 IOUT(I)=IABS(IN(I))
  3609              GO TO 200
  3610        C
  3611              ENTRY ISQDET(IN,IOUT)
  3612              NS= IN(N195)
  3613              DO 70 I=1,NS
  3614           70 IOUT(I)=IN(I)*IN(I)
  3615              GO TO 200
  3616        C
  3617              ENTRY ISGNAM(IN,IOUT)
  3618              NS=IN(N195)
  3619              DO 90 I=1,NS
  3620              IF(IN(I)) 91,92,93
  3621           91 IOUT(I)=-1
  3622              GO TO 90
  3623           92 IOUT(I)=0
  3624              GO TO 90
  3625           93 IOUT(I)=1
  3626           90 CONTINUE
  3627        C
  3628          200 IOUT(N195)=IN(N195)
  3629              IOUT(N194)=IN(N194)
  3630              IOUT(N193)=IN(N193)
  3631              RETURN
  3632              END
```

SUBROUTINE INGTOR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
INGTOR	LTI-4	409, 410
INGNCL	LTI-4	None

2. PURPOSE:

This subroutine is used to simulate a digital integrator.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FBCK	R	F	Feedback coefficient

4. CALLING SEQUENCES:

CALL INGTOR (DIN, DOUT)

Where: DIN contains the Input Waveform
DOUT contains the Output Waveform

The storage register (C1) is cleared before execution begins.

CALL INGNCL (DIN, DOUT)

Where: DIN contains the Input Waveform
DOUT contains the Output Waveform

The storage register (C1) is not cleared before execution begins.

5. RESTRICTIONS, REQUIREMENT, MISCELLANEOUS DATA

a. Flow Chart: Page 9-97

b. Cross Reference Table: Page 9-220

6. THEORY OF OPERATION

The block diagram of the digital integrator simulated by this module is shown in Figure INGTOR-1. The Z-plane transfer function is given by the following expression:

$$T(Z) = \frac{Z}{Z\text{-}FBCK}$$

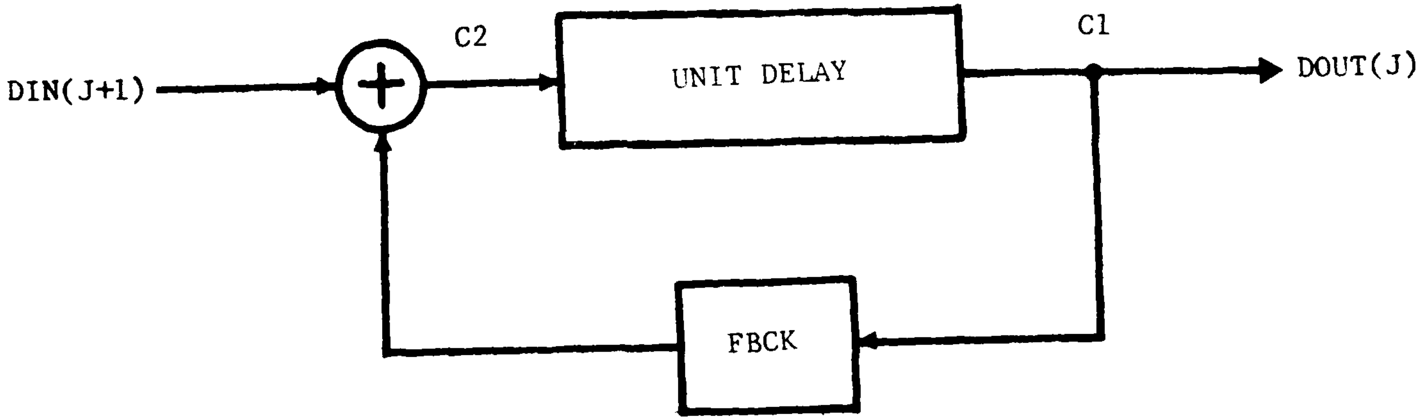


Figure INGTOR-1 BLOCK DIAGRAM OF INGTOR

```
      SUBROUTINE INGTCR(DIN,DOUT)
      COMMON/BK1/BK1(500)
      DIMENSION DIN(1),DOUT(1)
      EQUIVALENCE (BK1( 21), IDMY         ), (BK1( 75), FECK          )
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      C1 = 0.0
      ENTRY INGNCL(DIN,DOUT)
      N = IFCCC(DIN(N193))
      DOUT(N193) = DIN(N193)
      DOUT(N194) = DIN(N194)
      DOUT(N195) = DIN(N195)
      DO 10 J=1,N
      C2 = DIN(J)+C1*FECK
      DOUT(J) = C1
      C1 = C2
   10 CONTINUE
      RETURN
      END
```

SUBROUTINE IONOS

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
IONOS	LTI-1	511

2. PURPOSE:

This subroutine is used to simulate the effect of the ionosphere on signals propagating through it.

3. INPUT PARAMETERS:

Name	O/R	T	Description
RFFØ	R	F	Center frequency of the electromagnetic wave traversing the ionosphere
SEDENS	R	F	The integrated electron density along the propagation path

4. CALLING SEQUENCES:

```
CALL  IONOS (X,Y)
```

Where:

X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The effect of the earth's magnetic field is not included in this model.

b. The effect of electron collisions with ions and neutral particles is not included in this model. The major effect resulting from electron collisions is a damping term which results in absorption of energy from the electromagnetic wave transversing the ionized medium. For frequencies above 100 MHz electron collisions have negligible effect on the phase behavior of the ionosphere as a function of frequency.

c. Reference:

Berkowitz, R. S. ed. Modern Radar Analysis, Evaluation, and System Design, John Wiley & Sons, Inc., New York, 1965, pp 349-354, 364-370.

d. Flow Chart: Page 9-117

e. Cross Reference Table: Page 9-223

6. THEORY OF OPERATION

The index of refraction of an ionized medium is given by the following expression:

$$n = \sqrt{1 - \frac{N_e e^2}{m\pi f^2}} \qquad (1)$$

where: N_e is the electron density (electrons/cm^3)

e is the electron charge (4.8×10^{-10} esu)

f is the frequency of this incident electromagnetic energy

Collisions between electrons and ions or neutral particles are neglected.

The phase velocity of the propagating wave, V_p, is given by the following expression:

$$V_p = c/n \qquad (2)$$

where: c is the speed of light in free space

Therefore, combining equations 1 and 2 the following expression is obtained:

$$V_p = \frac{C}{\sqrt{1 - N_e e^2/m\pi f^2}} \tag{3}$$

The phase velocity will approach infinity as the denominator of equation 3 decreases to zero. When this condition arises, further wave propagation is impossible. The frequency for which this occurs is called the critical frequency and is given by the following expression:

$$f_c{}^2 = \frac{Ne\ e^2}{m\,\pi} = 8.0 \times 10^7\ N_e\ (\mathrm{Hz})^2 \tag{4}$$

Equation 3 now becomes the following:

$$V_p = \frac{C}{\sqrt{1 - f_c{}^2/f^2}} \tag{5}$$

The differential phase shift d$\emptyset$ that a wave will encounter in traversing an element of path length ds is given by the following expression:

$$d\emptyset = \frac{2\pi}{\lambda}\ ds \tag{6}$$

where: λ is the wavelength of the radiated electromagnetic energy

Since $\lambda = V_p/f$ the following expression is obtained by substitution of equation 5 into equation 6:

$$d\emptyset = \frac{2\pi f}{C}\sqrt{1 - fc^2/f^2}\quad ds \tag{7}$$

This expression can be further simplified by adding the term $\frac{fc^4}{4f^2}$ to the terms inside the square root radical. For most radar applications $f_c << f$ so this will introduce a very small error. Equation 7 now becomes:

$$d\phi = \frac{2\pi f}{C} \sqrt{\left(1 - \frac{f_c^2}{2f^2}\right)^2} \qquad ds$$

$$= \frac{2\pi f}{C} \left(1 - \frac{f_c^2}{2f^2}\right) \qquad ds \qquad (8)$$

In order to determine the phase shift over a propagation path from s_1 to s_2 the following expression will be used

$$\phi = \frac{2\pi f}{C} \int_{s_1}^{s_2} \left(1 - \frac{f_c^2}{2f^2}\right) \qquad ds$$

$$= \frac{2\pi f}{C} \int_{s_1}^{s_2} ds \quad - \frac{2\pi}{2Cf} \int_{s_1}^{s_2} f_c^2 \quad ds$$

substituting for f_c^2 the following is obtained:

$$\phi = \frac{2\pi f}{C} \int_{s_1}^{s_2} ds \quad - \frac{\pi}{Cf}\, 8.0 \times 10^7 \int_{s_1}^{s_2} N_e \; ds \qquad (9)$$

The first term of equation 9 is the linear phase shift due to propagation over a path of length $s_2 - s_1$ and will be omitted since it represents a time delay only

Therefore, the equation used to calculate the disperive effect of the ionosphere is given by the following expression:

$$\emptyset(f) = -2\pi \frac{4.0 \times 10^{7}}{3.0 \times 10^{10}} \frac{1}{f} \quad \text{SEDENS}$$

where: $\text{SEDENS} = \int_{s_1}^{s_2} N_e \, ds$ = integrated electron density

$$f = \text{RFF}\emptyset + k \, \Delta F, \quad -\frac{FEXT}{2\,\Delta F} < k < \frac{FEXT}{2\,\Delta F}$$

FEXT = frequency extent of input waveform representations

F = independent variable spacing of input waveform representations.

```
      SUBROUTINE IONOS(X,Y)
      COMMON/BLK1/ VAR(500)
      DIMENSION X(1),Y(1)
      EQUIVALENCE (VAR(145) ,SEDENS ) , (VAR(  3) , RFFO )
*   VARIABLE  SEDENS IS INTEGRATED ELECTRON DENSITY ALONG
*          PROPAGATION PATH  (ELECTRONS/CM*CM)
      DATA N193,N194,N195/-3,-2,-1/ CONS/ 1.3333E-03 /,PI2/6.2831853/
      NPTS=BOOL(X(N193))
      FREQ= (X(N194)+RFFO) *1.0E+09
      DELF=X(N195)*1.0E+09
      DO 200 J=1,NPTS
      THETA= CONS*SEDENS/FREQ
      THETA=(THETA-FLOAT(IFIX(THETA)))*PI2
      TMP=X(J)
      C=COS(THETA)
      S=SIN(THETA)
      X(J)=X(J)*C-Y(J)*S
      Y(J)=TMP*S+Y(J)*C
      FREQ=FREQ+DELF
  200 CONTINUE
      RETURN
      END
```

SUBROUTINE LAMPCP

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
LAMPCP	LTI-1 or LTI-4	458
LAMPRE	LTI-1 or LTI-4	456, 457

2. PURPOSE:

This subroutine is used to simulate a linear amplifier.

3. INPUT PARAMETERS:

Name	O/R	T	Description
GAIN	R	F	Amplifier gain

4. CALLING SEQUENCES:

CALL LAMPCP (XIN, YIN, XOUT, YOUT)

Where:
XIN contains the Input Waveform - R
YIN contains the Input Waveform - I
XOUT contains the Output Waveform - R
YOUT contains the Output Waveform - I

CALL LAMPRE (XIN, XOUT)

Where:
XIN contains the Input Waveform
XOUT contains the Output Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-68

b. Cross Reference Table: Page 9-217

6. THEORY OF OPERATION

The relationship between the input and output is given by the following expressions:

```
XOUT(J) = GAIN * XIN(J)  ;  LAMPCP & LAMPRE
YOUT(J) = GAIN * YIN(J)  ;  LAMPCP
```

482

```
   2027                SUBROUTINE LAMPCP(XIN,YIN,XOUT,YOUT)
   2028                DIMENSION XIN(1),YIN(1),XOUT(1),YOUT(1)
   2029                COMMON/BLK1/ E(200)
```

```
06/11/75        INPUT LISTING                     AUTOFLOW CHART SET - FWD/SCL  FADSIM

  CARD NO         ****                            CONTENTS                             ****

   2030                DATA N193,N194,N195/-3,-2,-1/
   2031                EQUIVALENCE  ( E(145) , GAIN )
   2032                MODE=1
   2033                GOTO 100
   2034                ENTRY LAMPRE(XIN,XOUT)
   2035                MODE=0
   2036          100   NPTS=LOCC(XIN(N193))
   2037                DO 200 J=1,NPTS
   2038                XOUT(J)=XIN(J)*GAIN
   2039          200   CONTINUE
   2040                IF(MODE.EQ.0) GOTO 500
   2041                DO 300 J=1,NPTS
   2042                YOUT(J)=YIN(J)*GAIN
   2043          300   CONTINUE
   2044                YOUT(N193)=YIN(N193)
   2045                YOUT(N194)=YIN(N194)
   2046                YOUT(N195)=YIN(N195)
   2047          500   XOUT(N193)=XIN(N193)
   2048                XOUT(N194)=XIN(N194)
   2049                XOUT(N195)=XIN(N195)
   2050                RETURN
   2051                END
```

SUBROUTINE MTIFLT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
MTIFLT (MODEDF=1)	LTI-4	430,431
MTINCL (MODEDF=1)	LTI-4	432,433
MTIIFT (MODEDF=2)	NL-2	430,431
MTIINC (MODEDF=2)	NL-2	432,433

2. PURPOSE:

This subroutine is used to simulate a double delay MTI processor. Either floating point or integer arithmetic is selectable.

3. INPUT PARAMETERS

a. MODEDF = 1

Name	O/R	T	Description
FFØ	O	F	Feed - forward coefficient - 0 delay (no delay)
FF1	O	F	Feed - forward coefficient - 1 delay (single delay)
FB1	O	F	Feedback coefficient - 1 delay
FB2	O	F	Feedback coefficient - 2 delay (double delay)

b. MODEDF=2

Name	O/R	T	Description
IFFØN	O	I	Numerator of Feed - forward coef - 0 delay
IFFØD	R	I	Denominator of feed - forward coef- 0 delay
IFF1N	O	I	Numerator of feed - forward coef - 1 delay

Name	O/R	T	Description
IFF1D	R	I	Denominator of feed - forward coef - 1 delay
IFB1N	O	I	Numerator of feedback coef - 1 delay
IFB1D	R	I	Denominator of feedback coef - 1 delay
IFB2N	O	I	Numerator of feedback coef - 2 delay
IFB2D	R	I	Denominator of feedback coef. - 2 delay
NBITS	R	I	Number of bits to be used in storing one range sample in the arrays (ISR1 and ISR2) which represents the digital delay lines (includes sign bit).

4. CALLING SEQUENCES:

Floating point arithmetic

CALL MTIFLT (DIN, DOUT)

Where: DIN contains the Input Waveform

DOUT contains the Output Waveform

The range bit storage arrays (SR1 and SR2) are cleared prior to execution.

The entry point is used only if MODEDF=1.

Floating point arithmetic

CALL MTINCL (DIN, DOUT)

Where: DIN contains the Input Waveform

DOUT contains the Output Waveform

The range bin storage arrays (SR1 and SR2) are not cleared prior to execution except for the first execution of this subroutine. This entry point is used only if MODEDF=1.

Integer arithmetic

CALL MTIIFT (IN, IOUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

The range bin arrays (ISR1 and ISR2) are cleared prior to execution.

This entry point is used only if MODEDF=2.

Integer Arithmetic

CALL MTIINC (IN, OUT)

Where: IN contains the Input Waveform

IOUT contains the Output Waveform

The range bin storage arrays (ISR1 and ISR2) are not cleared prior to execution except for the first execution of this subroutine.

This entry point is used only if MODEDF=2.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The maximum number of range bins simulated is 2048. If more than 2048 range samples are contained in the input array, only the first 2048 are processed.

b. The parameter MODEDF is used by the simulation controller to determine whether the floating point or the fixed point module entry points are to be used.

c. The entry points which use integer arithmetic are considered nonlinear because of truncation in arithmetic and saturation in the delay line.

d. Flow Chart: Page 9-201

e. Cross Reference Table: 9-234

6. THEORY OF OPERATION

The block diagram of the two-delay MTI processor digital filters simulated by this module are shown in Figure MTIFLT-1 and MTIFLT-2. The Z-plane transfer function for each range bin (floating point arithmetic) is given by the following expression:

$$T(z) = \mathrm{FF}\emptyset \; \frac{Z^2 + \dfrac{\mathrm{FF1}}{\mathrm{FF}\emptyset} Z + \dfrac{1}{\mathrm{FF}\emptyset}}{Z^2 - \mathrm{FB1}\, Z - \mathrm{FB2}}$$

The Z-plane transfer function for each range bin (integer arithmetic) is approximated by the following expression:

$$T(Z) = \frac{\mathrm{IFF}\emptyset\mathrm{N}}{\mathrm{IFF}\emptyset\mathrm{D}} \; \frac{Z^2 + \dfrac{\mathrm{IFF1N} \cdot \mathrm{IFF}\emptyset\mathrm{D}}{\mathrm{IFF}\emptyset\mathrm{N} \cdot \mathrm{IFF1D}} Z + \dfrac{\mathrm{IFF}\emptyset\mathrm{D}}{\mathrm{IFF}\emptyset\mathrm{N}}}{Z^2 - \dfrac{\mathrm{IFB1N}}{\mathrm{IFB1D}} Z - \dfrac{\mathrm{IFB2N}}{\mathrm{IFB2D}}}$$

The delay represented by the Z operator is determined by the radar pulse repetition interval.

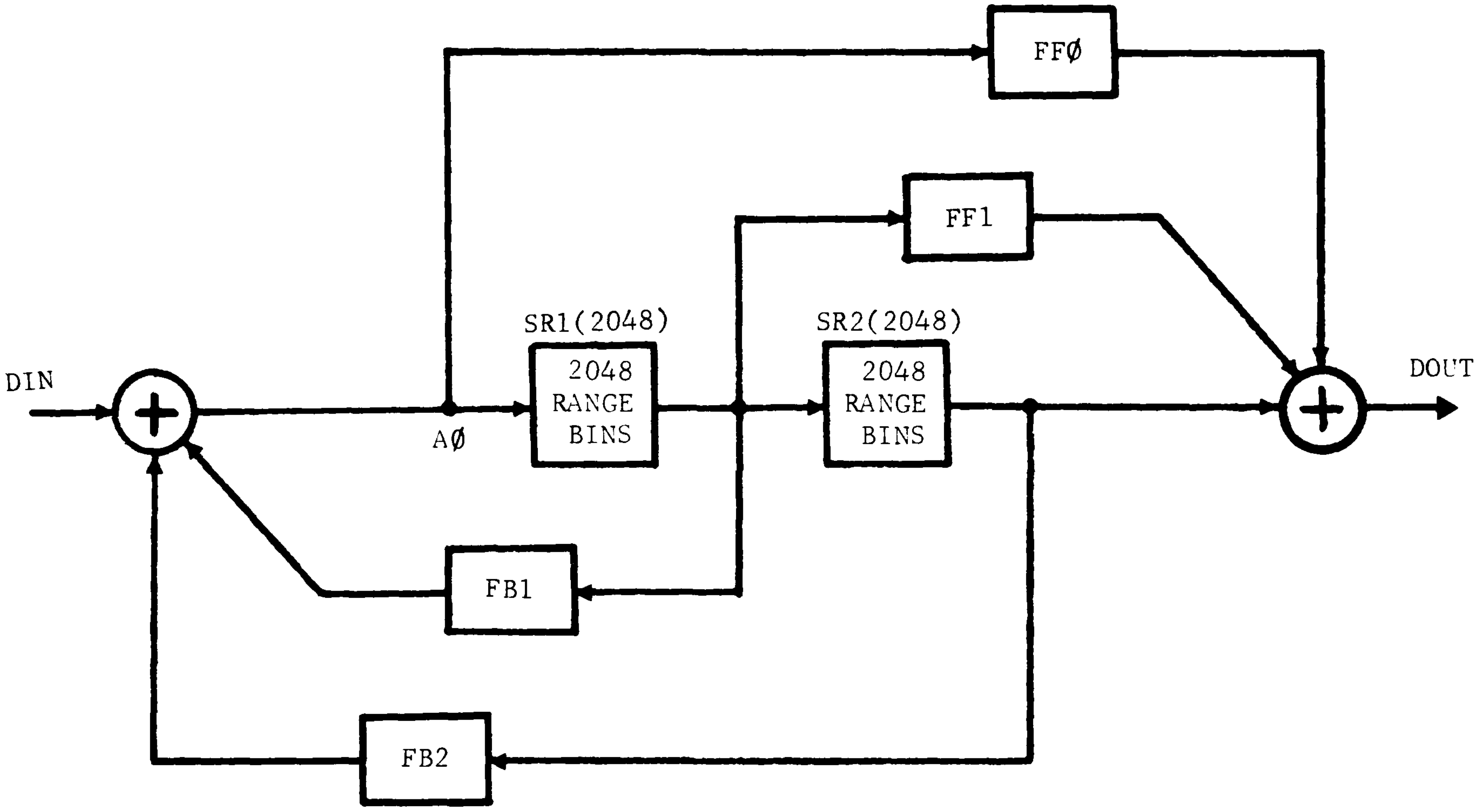


Figure MTIFLT-1 BLOCK DIAGRAM OF MTIFLT/MTINCL
(Floating-point Arithmetic)

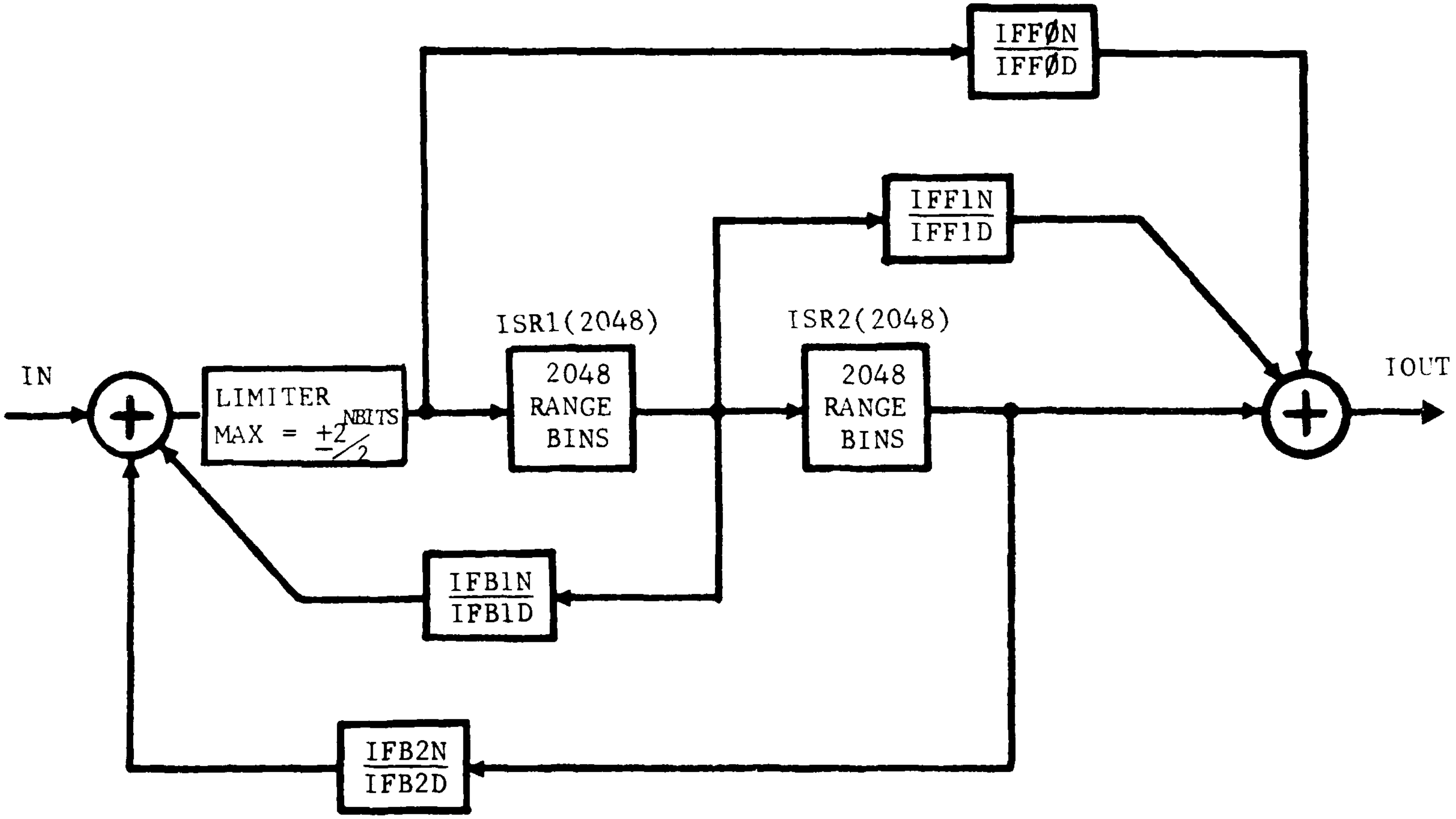


Figure MTIFLT-2 BLOCK DIAGRAM OF MTIFLT/MTINCL (Integer Arithmetic)

```
      SUBROUTINE MTIFLT(DIN,DOUT)                                       UC7TS001
      COMMON/BLK1/BK1(500)                                              UC7TS002
      DIMENSION DIN(1),DOUT(1),IN(1),IOUT(1),SK1(2048),SK2(2048),       UC7TS003
     * ISK1(2048),ISK2(2048)                                            UC7TS004
      EQUIVALENCE (SK1(1),ISK1(1) ) , (SK2(1),ISK2(1) )                 UC7TS005
      DATA N193,N194,N195,N196/-3,-2,-1,0/                              UC7TS006
      DATA IFLG/0/                                                           320
     0EQUIVALENCE (BK1( 21), IDMY      ), (BK1( 68), FF0       ),   UC7TS008
     1            (BK1( 69), FF1       ), (BK1( 70), FB1       ),   UC7TS009
     2            (BK1( 71), FB2       )                            UC7TS010
      EQUIVALENCE  (  BK1(160), IFF0N      ),(  BK1(161), IFF0D    ),   UC7TS011
     *             (  BK1(162), IFF1N      ),(  BK1(163), IFF1D    ),   UC7TS012
     *             (  BK1(164), IFB1N      ),(  BK1(165), IFB1D    ),   UC7TS013
     *             (  BK1(166), IFB2N      ),(  BK1(167), IFB2D    ),   UC7TS014
     *             (  BK1(169), NBITS      )                            UC7TS015
      ICON=1                                                                 401
      GO TO 40                                                               402
      ENTRY MTINCL(DIN,DOUT)                                                 403
      ICON=2                                                                 404
   40 IF(IFLG.EQ.1.AND.ICON.EQ.2) GO TO 50                                   405
      DO 20 J=1,2048                                                    UC7TS016
```

```
 CARD NO      ****                          CONTENTS                            ****

   5220             SR1(J)=0.0                                             UC7TS017
   5221             SR2(J)=0.0                                             UC7TS018
   5222          20 CONTINUE                                               UC7TS019
   5223             IFLG=1                                                      450
   5224          50 CONTINUE                                                    451
   5225             N = IBOOL(DIN(N193))                                   UC7TS021
   5226             IF(N.LE.2048) GO TO 15                                 UC7TS022
   5227             N=2048                                                 UC7TS023
   5228             WRITE(6,25)                                            UC7TS024
   5229          25 FORMAT( ' TOO MANY  POINTS IN INPUT ARRAY....FIRST 2048 PROCESSED'UC7TS025
   5230            * )                                                     UC7TS026
   5231          15 DOUT(N193) = BOOL(N)                                   UC7TS027
   5232             DOUT(N194) = DIN(N194)                                 UC7TS028
   5233             DOUT(N195) = DIN(N195)                                 UC7TS029
   5234             DO 10 J=1,N                                            UC7TS030
   5235             A0= SR1(J)*FB1+SR2(J)*FB2+DIN(J)                       UC7TS031
   5236             DOUT(J)= A0*FF0+SR1(J)*FF1+ SR2(J)                     UC7TS032
   5237             SR2(J)=SR1(J)                                          UC7TS033
   5238             SR1(J)=A0                                              UC7TS034
   5239          10 CONTINUE                                               UC7TS035
   5240             RETURN                                                 UC7TS036
   5241             ENTRY MT11FI(IN,IOUT)                                  UC7TS037
   5242             ICON=1                                                      621
   5243             GO TO 60                                                    622
   5244             ENTRY MT11NC(IN,IOUT)                                       623
   5245             ICON=2                                                      624
   5246          60 IF(IFLG.EQ.1.AND.ICON.EQ.2) GO TO 70                        625
   5247             DO 100 J=1,2048                                        UC7TS038
   5248             ISR1(J)=0                                              UC7TS039
```

4-90a

```
      ISR2(J)=0                                              UC7TS040
  100 CONTINUE                                               UC7TS041
      IFLG=1                                                      670
   70 CONTINUE                                                    671
      IF(NBITS.GT.0.AND.NBITS.LE.31) GO TO 95                UC7TS043
      NBITS=31                                               UC7TS044
   95 CONTINUE                                               UC7TS045
      MAX=2**(NBITS-1)                                       UC7TS046
      N=IN(N193)                                             UC7TS047
      IF(N.LE.2048) GO TO 105                                UC7TS048
      N=2048                                                 UC7TS049
      WRITE(6,25)                                            UC7TS050
  105 IOUT(N193)= N                                          UC7TS051
      IOUT(N194)=IN(N194)                                    UC7TS052
      IOUT(N195)=IN(N195)                                    UC7TS053
      DO 110 J=1,N                                           UC7TS054
      IO=(ISR1(J)*IFB1N)/IFB1D +(ISR2(J)*IFB2N)/IFB2D + IN(J)   UC7TS055
      IF(IABS(IO).GT.MAX) IO=ISIGN(MAX,IO)                   UC7TS056
      IOUT(J)=(IO*IFF0N)/IFF0D +(ISR1(J)*IFF1N)/IFF1D + ISR2(J)   UC7TS057
      ISR2(J)=ISR1(J)                                        UC7TS058
      ISR1(J)=IO                                             UC7TS059
  110 CONTINUE                                               UC7TS060
      RETURN                                                 UC7TS061
      END
```

SUBROUTINE NONLIN

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
NONLIN	NL-1	401, 402

2. PURPOSE:

This subroutine is used to simulate a zero memory nonlinear device.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NPTS	R	I	Number of points used to specify the transfer function
TFN	R	F	Array containing the specification of the transfer function. The coefficients for the Jth point are the following: TFN(1,J) = Input voltage TFN(2,J) = Output voltage

4. CALLING SEQUENCES:

CALL NONLIN (A,$mmmm)

Where:

A contains the Input Waveform

A contains the Output Waveform

$mmmm is the statement number in the calling program to which control is returned if a discrepancy is detected in the input data.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The independent variable in the transfer function specification array must be monotonically increasing.

b. The input data must be within the limits of the specified transfer function.

c. Flow Chart: Page 9-135

d. Cross Reference Table: Page 9-225

6. THEORY OF OPERATION

The input vs output characteristics of the device to be simulated are specified via the array TFN. A table lookup and linear interpolation scheme are used to compute the output value for each sample of the input waveform. An abnormal termination will occur if an input sample is outside the range of input values specified in the array TFN.

An example of the input data for this module is the nonlinear transfer function shown in Figure NONLIN-1. The associated input data statement is as follows:

```
$NL401 TFN = -2.0, -2.0, -1.0, -2.0, 1.0, 2.0, 2.0,
2.0, NTFN = 4$
```

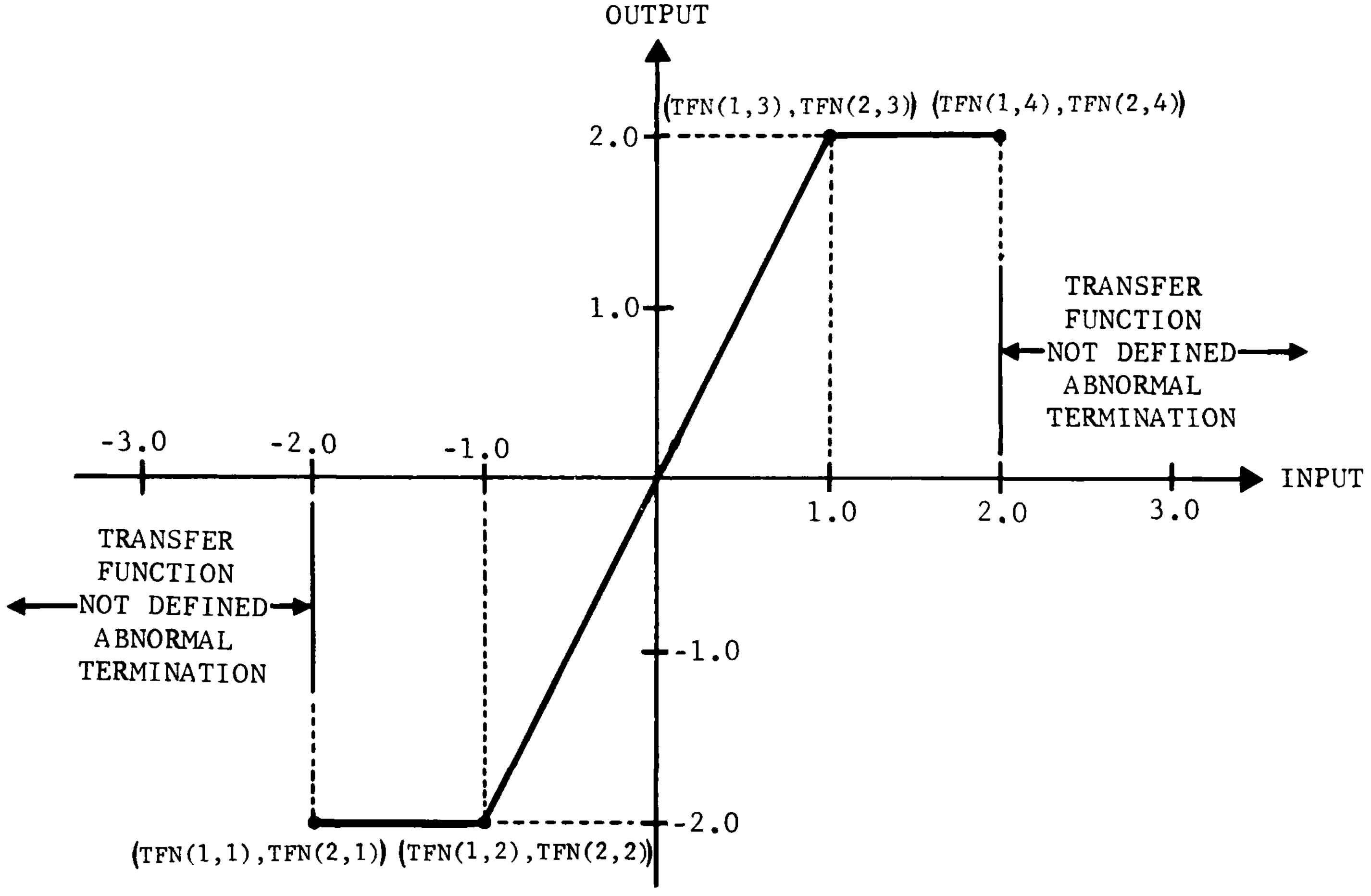


Figure NONLIN-1 EXAMPLE OF NON-LINEAR TRANSFER FUNCTION

```
      SUBROUTINE NONLIN(A,*)
      COMMON/BLK1/BK1(500)
      DIMENSION A(1),TFN(2,50)
      EQUIVALENCE (BK1( 21), IDMY        ), (BK1( 67), NPTS        ),
     *            (BK1(201), TFN(1,1)    )
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      NIN = IFOOL(A(N193))
      K=1
      DO 100 J=1,NIN
      GO TO 105
  103 K=K+1
      GO TO 105
  104 K=K-1
  105 IF(K.GT.NPTS) GO TO 500
      IF(K.LT.1) GO TO 500
      IF(A(J).LT.TFN(1,K))GO TO 104
      IF(A(J).GT.TFN(1,K+1))GO TO 103
      PCT =  (A(J)-TFN(1,K)) /(TFN(1,K+1)-TFN(1,K))
      A(J) = TFN(2,K) + PCT*(TFN(2,K+1)-TFN(2,K))
  100 CONTINUE
      RETURN
  500 CONTINUE
      RETURN 1
      END
```

SUBROUTINE PHDEC

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PHDEC	LTI-2	508, 509

2. PURPOSE:

This subroutine is used to simulate an analog phase decoder.

3. INPUT PARAMETERS

a. Automatic mode; INPTF = 0

Name	O/R	T	Description
SIMFØ	R	F	Center frequency of waveform previously generated by FGENXY/FGENMP. This parameter is used to specify the center frequency of the tapped delay line.
INPTF	R	I	Set = 0 for this mode
SPW	R	F	Subpulsewidth of waveform previously generated by FGENXY/FGENMP. This parameter is used to specify the delay line tap spacing (TAPSPC).
XMITPC	R	F	Array containing the phase code of the waveform generated by FGENXY/FGENMP.
NSUBP	R	I	Number of subpulses in the waveform generated by FGENXY/FGENMP. This parameter specifies the number of delay line taps.

NOTE: No user supplied inputs required for this mode. All required parameters are initialized by FGENXY/FGENMP.

b. User Supplied data mode; INPTF = 1

Name	O/R	T	Description
FØDEC	R	F	Center frequency of the tapped delay line.
TAPSPC	O	F	Spacing between delay line taps. This parameter is set equal to SPW if it is not specified by the user.
INPTF	R	I	Set = 1 for this mode
XMITPC	R	F	Array containing the phase code of the waveform generated by FGENXY/FGENMP. (Initialized by FGCENXY/FGENMP)
NSUBP	R	I	Number of subpulses in the waveform generated by FGENXY/FGENMP. This parameter specifies the number of delay line taps. (Initialized by FGENXY/FGENM)

4. CALLING SEQUENCES:

CALL PHDEC (X,Y)

Where: X contains the Output Waveform - R
Y contains the Output Waveform - I

5. RESTRICTIONS, RECOMMENDATIONS, MISCELLANEOUS DATA

a. If INPTF=1 and TAPSPC is specified, then TAPSPC should be an integral multiple of the period of the waveform center frequency, i.e. TAPSPC = N*1.0/SIMFØ where N is an integer. Otherwise, the taps will not have the desired phase shifters as specified in the array XMITPC.

b. Flow Chart: Page 9-78

c. Cross Reference Table: Page 9-218

6. THEORY OF OPERATION

This module generates the transfer function of a tapped analog delay line. The main use of this module is to simulate a surface acoustic wave device. A block diagram of this device represented by this module is shown in Figure PHDEC-1. This module is structured to represent only the taps (energy pickoffs) of the delay line. The wave launches must be specified separately in terms of its transfer function. This can be done via either the FILT or the WEIT modules. A pictorial diagram of a surface acoustic wave device and its representation are shown in Figures PHENC-2(a) and -2(b). The impulse response of the device represented by this module is given by the following expression:

$$h(t) = \sum_{K=1}^{K=NSUBP} g_K \, \delta(t-Kt_d) \, e^{j\phi_K}$$

where:

g_K = tap gain

t_d = tap spacing

ϕ_K = tap phase shift

The transfer function computed by this module is given by the following expression:

$$H(f) = \mathcal{F}[h(t)] = \int_{-\infty}^{\infty} \sum_{k=1}^{NSUBP} g_k \, \delta(t-kt_d) \, e^{j\phi k} e^{-j2\pi ft} \, dt$$

Interchanging the order of integration and summation

$$H(f) = \sum_{1}^{NSUBP} g_k e^{j\phi_k} \int_{-\infty}^{\infty} \delta(t-kt_d) \, e^{-j2\pi ft} \, dt$$

$$H(f) = \sum_{1}^{NSUBP} g_k e^{j\phi_k} \, e^{-j2\pi fkt_d}$$

This version of the phase decoder module sets all $g_k = 1.0$. If a phase decoder with gain weighting or errors is desired then the DFT routine should be used to compute the transfer function.

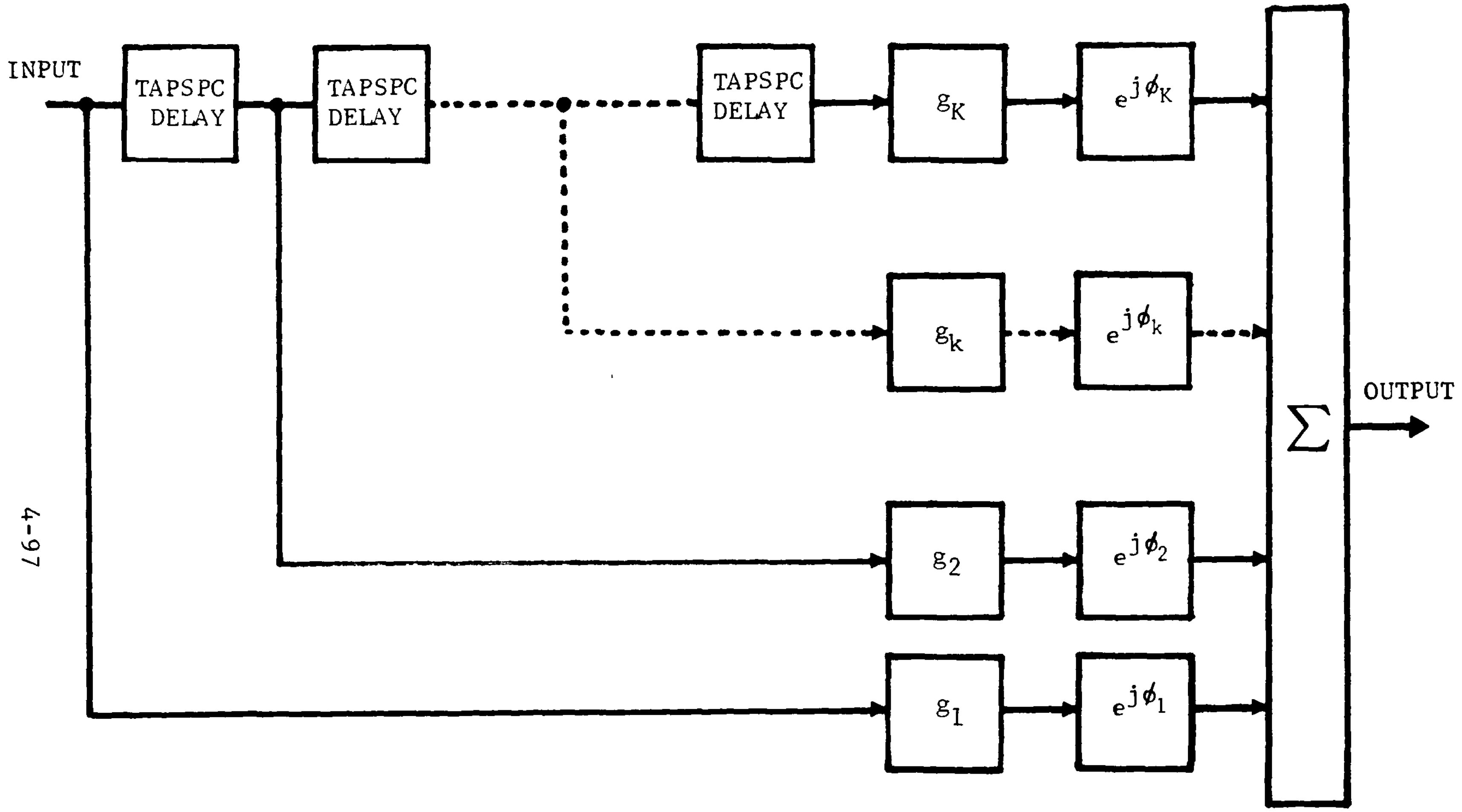


Figure PHDEC-1 BLOCK DIAGRAM OF A TAPPED DELAY LINE PHASE DECODER

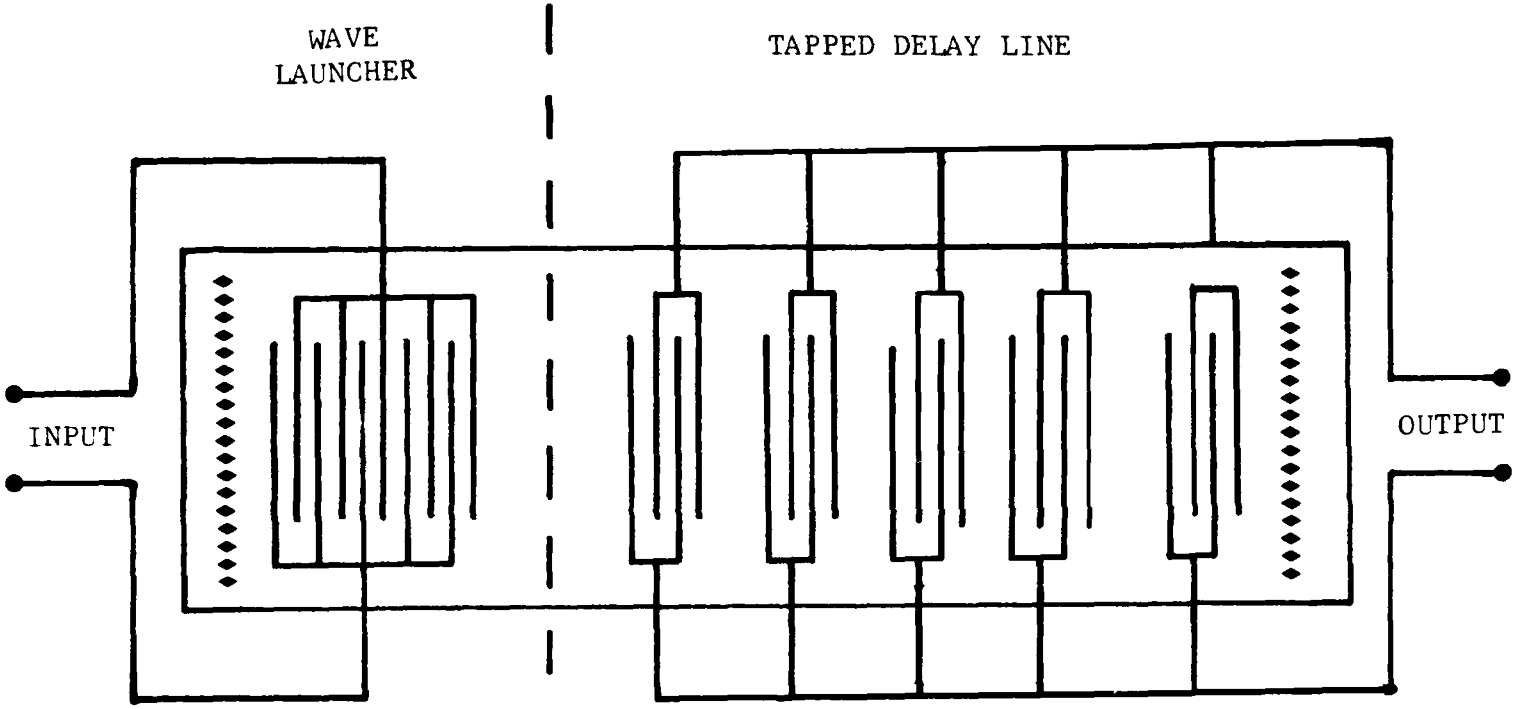


Figure PHDEC-2a SURFACE WAVE DEVICE PICTORIAL DIAGRAM

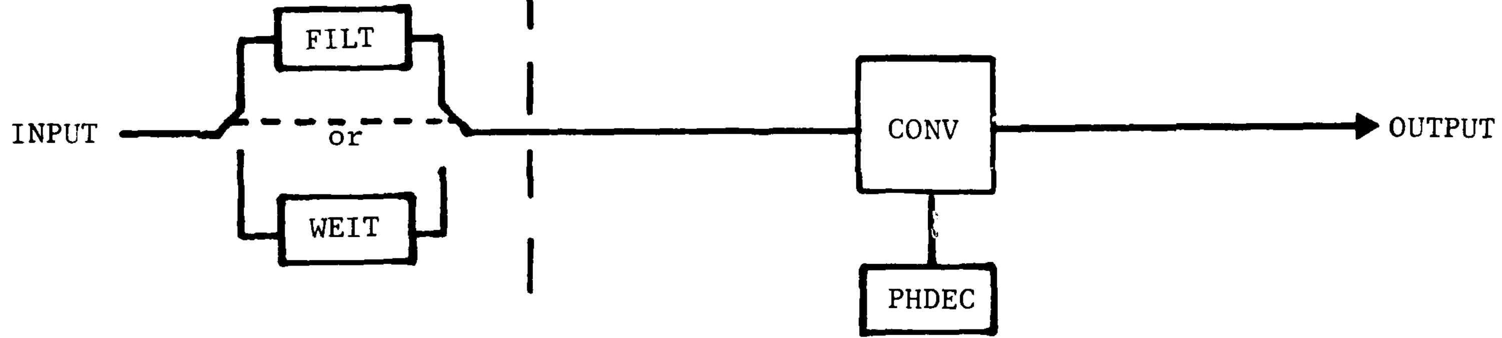


Figure PHDEC-2b SIMULATION MODEL BLOCK DIAGRAM

```
              SUBROUTINE PREDC(X,Y)
              DIMENSION X(1),Y(1)
              COMMON/PREDCT/ XNITPC(300),NSUBP,SPW
              COMMON/SERI/ ITEMP(199),NIMP,DIN(3,100)
              EQUIVALENCE  (ITEMP(102),  FODEC) , (ITEMP(154),  TAPSPC),
             *                   (ITEMP(  1),  SIMPO) , (ITEMP(155),  INPTF)
              IF(INPTF.EQ.1) SIMPO=FODEC
              IF(INPTF.EQ.0) TAPSPC=SPW
              IF(INPTF.EQ.1.AND.TAPSPC.EQ.0.0) TAPSPC=SPW
              IPASS=NSUBP
              TIME=0.0
        100   CONTINUE
              NIMP=IPASS
              IF(IPASS.GT.100) NIMP=100
              DO 200 J=1,NIMP
              DIN(1,J)=XNITPC(IPASS+1-J)
              DIN(2,J)=TIME
              DIN(3,J)=1.0
              TIME=TIME+TAPSPC
        200   CONTINUE
              IF(IPASS.EQ.NSUBP) CALL DFTPD(X,Y)
              IF(IPASS.NE.NSUBP) CALL DFTNCL(X,Y)
              IF(NIMP.NE.100) RETURN
              IPASS=IPASS-100
              IF(IPASS.EQ.0) RETURN
              GOTO 100
              END
```

SUBROUTINE PHENC

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PHENC/FGGNXY	SOU-1 or LTI-3	506
PHENC/FGENMP	SOU-1 or LTI-3	507

2. PURPOSE:

This subroutine generates a phase code which is subsequently used by FGENXY/FGENMP to generate a binary phase coded waveform. Either the module FGENXY or FGENMP is automatically scheduled for execution.

3. INPUT PARAMETERS

a. MODEPH = 1; Barker Phase Code

Name	O/R	T	Description
MODEPH	R	I	Set = 1 for this mode
NSUBP	R	I	Number of subpulses to be generated. The allowable values of this parameter and the corresponding Barker codes are as follows: 2 + - 3 + + - 4 + + - + 5 + + + - + 7 + + + - - + - 11 + + + - - - + - - + - 13 + + + + + - - + + - + - +

b. MODEPH = 2; Pseudo Random Phase Code

Name	O/R	T	Description
MODEPH	R	I	Set = 2 for this mode
NSUBP	R	I	Number of subpulses. Maximum value = 300 for this mode.

AD-A102 278   GENERAL DYNAMICS FORT WORTH TEX CONVAIR AEROSPACE DIV   F/G 17/9
ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING. VOLUME II. PAR--ETC(U)
JUN 76 R J HANCOCK, F H CLEVELAND   F30602-73-C-0380
UNCLASSIFIED   RADC-TR-76-186-VOL-2-PT-1   NL

3 of 4
AD
A102278

Name	O/R	T	Description
IPY	R	I	Array representing the shift register used to generate the pseudo random sequence. The storage cells of this shift register contains either 1 or 0. The user can specify the initial condition of this register if desired.
NSR	R	I	Number of stages in the shift register.

c. MODEPH = 3; User Specified Phase Code

Name	O/R	T	Description
MODEPH	R	I	Set = 3 for this mode
NSUBP	R	I	Number of subpulses. For this mode this parameter is limited to 35.
CODE	R	B	Word containing the phase code. Each bit of this 36 bit word represents the phase of one subpulse. The phase for the first subpulse is determined by the LSB; i.e. right most bit.

4. CALLING SEQUENCES:

CALL PHENC ($mmmm)

Where: mmmm is the statement number to which control will be transferred if an error is detected in the input data.

5. RESTRICTIONS, RECOMMENDATIONS, MISCELLANEOUS DATA

a. Reference for Barker and Pseudo random phase codes:

Skolnik, M. I.: RADAR HANDBOOK, McGraw-Hill Book Company, New York, 1970, pp 20-19 and 20-20.

b. Flow Chart: Page 9-167

c. Cross Reference Table: Page 9-229

6. THEORY OF OPERATION

This module generates a phase code determined by the input parameters and loads the array PCODE with the phase code. The operations performed for MODEPH = 1 or 3 consist of connecting bits of the word CODE into phase angles. A binary "1" is converted into 180 degrees and a binary "0" is converted into 0 degrees.

Figure PHDEC-1 is a block diagram of the shift registers used to generate the pseudo random phase codes. The parameter NSR determines the selection of shift register taps (Reference Skolnik, p 20-20, table 6). The output from the delay line is converted to a phase code according to the procedure outlined in the preceeding paragraph.

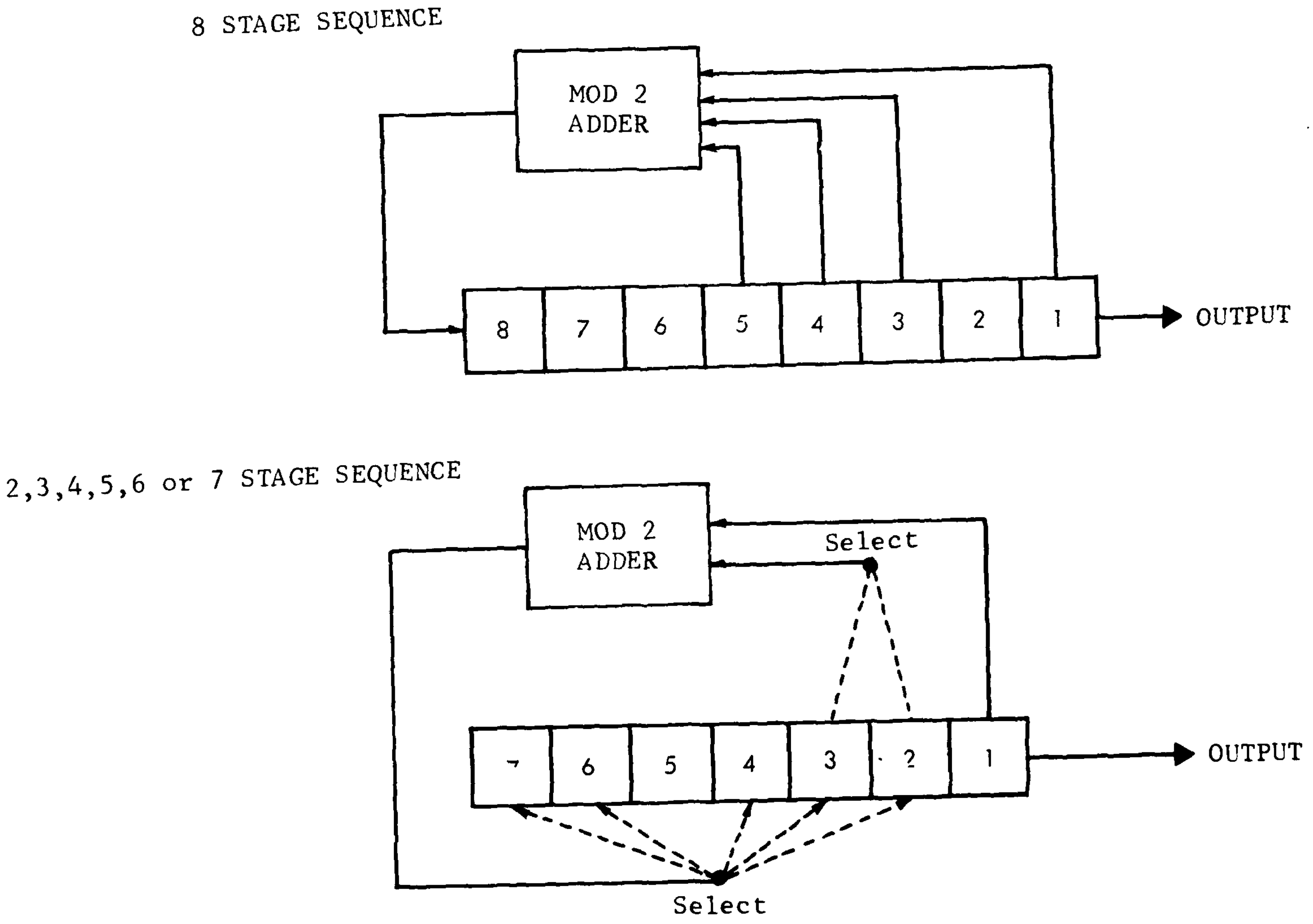


Figure PHENC-1 BLOCK DIAGRAM OF PSEUDO-RANDOM PHASE CODE GENERATORS

```
      SUBROUTINE PHENC(*)
      COMMON/BLK27/ B(200),PCODE(300)
      EQUIVALENCE  (B( 96),  NSUBP), (B(183), MODEPH),
     *             (B(184),  NSP   ) , (B(186), IPY(1) )  ,
     *             (B(194),  CODE  )
      DIMENSION IPY(5),BCODE(13)
      DATA BCODE/[illegible]/,N1/1/
      IF(MODEPH.EQ.2)CODE=BCODE(NSUBP)
      IF(MODEPH.EQ.1.OR.MODEPH.EQ.3) GOTO 100
      IT1=2
      IF(NSP.EQ.3.OR.NSP.EQ.5) IT1=3
      DO 300 J=1,NSUBP
      ISUM=IPY(1)+IPY(IT1)
      IF(NSP.NE.5) GOTO 150
      ISUM=ISUM+IPY(4)+IPY(5)
      IF(ISUM.EQ.4) ISUM=0
      IF(ISUM.EQ.3) ISUM=1
  150 IF(ISUM.EQ.2) ISUM=0
      IPY(NSP+1)=ISUM
      PCODE(J)=10.0*IPY(1)
      DO 200 K=1,NSP
      IPY(K)=IPY(K+1)
  200 CONTINUE
  300 CONTINUE
      RETURN
  100 CONTINUE
      IF(NSUBP.GT.5 ) RETURN 1
      K=[illegible]-NSUBP
      DO 400 J=1,NSUBP
      IPH=FLD(K+J,1,CODE)
      PCODE(J)=IPH*10.0
  400 CONTINUE
      RETURN
      END
```

SUBROUTINE RDIGFL

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
RDIGFL	LTI-4	405
RDFNCL	LTI-4	406

2. PURPOSE:

This subroutine is used to simulate a double delay digital filter and is capable of synthesizing a filter transfer function having two poles and two zeros.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FFØ	0	F	Feed - forward coefficient - 0 delay
FF1	0	F	Feed - forward coefficient - 1 delay
FB1	0	F	Feedback coefficient - 1 delay
FB2	0	F	Feedback coefficient - 2 delay

4. CALLING SEQUENCES:

CALL RDIGFL (X,Y)

Where:
X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

The storage registers (RR1, RR2, RI1, and RI2) are cleared before execution begins.

CALL RDFNCL (X,Y)

Where:
X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the output Waveform - I

The storage registers are not cleared before execution begins.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-100

b. Cross Reference Table: Page 9-221

6. THEORY OF OPERATION

The block diagram of the two delay digital filter simulated by this module is shown in Figure RDIGFL-1. The Z-plane transfer function is given by the following expression:

$$T(Z) = FF\emptyset \frac{Z^2 + \frac{FF1}{FF\emptyset} Z + {}^{1}/_{FF\emptyset}}{Z^2 - FB1\ Z - FB2}$$

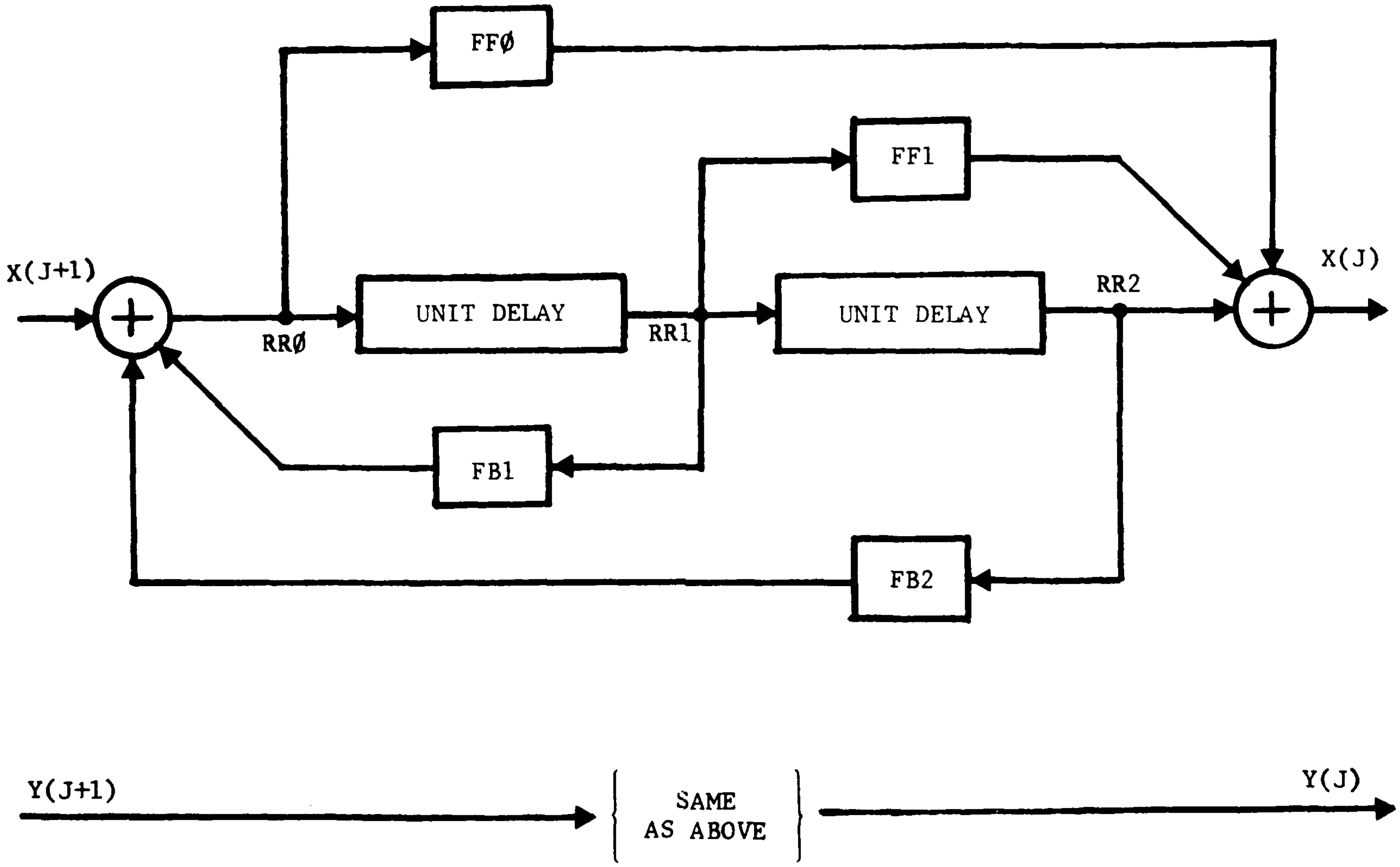


Figure RDIGFL-1   BLOCK DIAGRAM OF RDIGFL/RDFNCL

```
      SUBROUTINE RDIGFL (X,Y)
      COMMON/BLK1/BK1(500)
      EQUIVALENCE (BK1(68), FF0 ) , (BK1(69), FF1 ) , (BK1(70), FB1 ),
     *             (BK1(71), FB2 )
      DATA N143/-3/
      DIMENSION X(1),Y(1)
      RR1=0.0
      RR2=0.0
      RI1=0.0
      RI2=0.0
      ENTRY RDFNCL (X,Y)
      N=BOOL(X(N143))
      DO  10 J=1,N
      RR0=RR1*FB1+RR2*FB2+X(J)
      RI0=RI1*FB1+RI2*FB2+Y(J)
      X(J)=RR0*FF0+RR1*FF1+RR2
      Y(J)=RI0*FF0+RI1*FF1+RI2
      RR2=RR1
      RR1=RR0
      RI2=RI1
      RI1=RI0
   10 CONTINUE
      RETURN
      END
```

SUBROUTINE RNDARY

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
RNDARY	SOU-1	214, 215
ADDRND	SOU-1S	235, 236
ADRNDC	SOU-1S	237

2. PURPOSE:

This subroutine loads an array with random numbers generated by the random number generator function subroutine RRAND.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NPTS	R	I	Number of random distribution samples to be loaded into the array (RNDARY only).
NTYPE	R	I	Code indicating the type of random distribution to be used (RNDARY and ADDRND only).
TI	R	F	Simulation sampling increment (RNDARY only).

4. CALLING SEQUENCES:

CALL RNDARY(RND)

Where: RND contains the Output Waveform

CALL ADDRND(RND)

Where: RND contains the Input Waveform

RND contains the Output Waveform

CALL ADRNDC(RND,RNDY)

Where: RND contains the Input Waveform - R

RNDY contains the Input Waveform - I

RND contains the Output Waveform - R

RNDY contains the Output Waveform - I

This module is used only to add Gaussian samples to the input waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. User supplied function subprogram RRAND is called by the subprogram and must be initialized via NL101 or NL102 before this subroutine is executed.

b. The random distribution code (NTYPE) is as follows:

(1) Floating point uniform distribution with specified mean and extent.

(2) Floating point Rayleigh distribution with specified standard deviation.

(3) Floating point Gaussian distribution with specified mean and standard deviation.

(4) Positive integer uniform distribution from 0 to 2**36.

(5) Not used by this subprogram.

(6) Swerling Target Models #1 & #2 with specified mean cross section.

(7) Swerling Target Models #3 & #4 with specified mean cross section.

(8) Sine distribution.

c. Flow Chart: Page 9-150

d. Cross Reference Table: Page 9-227

6. THEORY OF OPERATION

The mechanization equations for each entry point are the following:

RNDARY: RND(J) = RRAND (NTYPE) ; KJ ≤ NPTS

ADDRND: RND(J) = RND(J) + RRAND (NTYPE); the range of J is determined by the input array.

ADRNDC: RND(J) = RND(J) + RRAND(3)
RNDY(J) = RNDY(J) + DUM

where:	RRAND(3)	contains the cosine gaussian distribution sample
	DUM	contains the sine gaussian distribution sample

```
       SUBROUTINE RNDARY(RND)
      COMMON/BLK1/BK1(500)                                          R
       COMMON/BLKRND/ RNDDAT(141)
      DIMENSION RND(1),RNDY(1)
       EQUIVALENCE (BK1( 44), NPTS  ),(BK1( 45), NTYPE)
       EQUIVALENCE (BK1( 12), T1)
      DATA N193,N194,N195,N196/-3,-2,-1,0/                          R
C                                                                   R
       DO 10 I=1,NPTS
       RND(I)=RRAND(NTYPE)
  10   CONTINUE
       RND(N193)=BOOL(NPTS)
       RND(N194)=0.0
       RND(N195)=T1
       RETURN
       ENTRY ADDRND(RND)
      NPTS=BOOL(RND(N193))                                         RN
      DO 20 I=1,NPTS                                                R
      RND(I)=RND(I)+RRAND(NTYPE)                                    R
   20 CONTINUE                                                      R
C                                                                   R
      RETURN                                                        R
       ENTRY ADRNDC(RND,RNDY)
       NPTS=BOOL(RND(N193))
       DO 40 I=1,NPTS
       RND(I)=RND(I)+RRAND(NTYPE)
       RNDY(I)=RNDY(I)+RNDDAT(9)
  40   CONTINUE
       RETURN
      END
```

SUBROUTINE SHIFT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
SHIFT	LTI-1	224,225
SHIFTS	LTI-1S	226,227
RSHIFT	LTI-1	229,230
RSHFTS	LTI-1S	231,232

2. PURPOSE:

This subroutine is used to delay a waveform in time and/or introduce a constant phase shift.

3. INPUT PARAMETERS:

Name	O/R	T	Description
TØ	R	F	Time delay
THT	R	F	Phase shift
TJIT	O	F	Time delay jitter
SIMFØ	R	F	Center frequency to be used in computing time delay effect.

4. CALLING SEQUENCES:

CALL SHIFT (X,Y,A,B)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
A contains the Output Waveform - R
B contains the Output Waveform - I

The basic mechanization equation for this entry point is the following:

$$A(J)+jB(J)=\Big[X(J)+jY(J)\Big]*e^{j\left[\text{THT}-2\pi\ (\text{FSTRT}+\text{FI}*J)\text{T}\emptyset\right]}$$

Where: FI = frequency increment
FSTRT = frequency corresponding to the first element of the input array

CALL SHIFTS (X,Y,A,B)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
A contains the Output Waveform - R
B contains the Output Waveform - I

The basic mechanization equation for this entry point is the following:

$$A(J)+jB(J)=A(J)+jB(J)+\Big[X(J)+jY(J)\Big]*e^{j\left[\text{THT}-2\pi(\text{FSTRT}\ +\text{FI}*J)\text{T}\emptyset\right]}$$

CALL RSHIFT (X,Y,A,B)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
A contains the Output Waveform - R
B contains the Output Waveform - I

The basic mechanization equation for this entry point is the following:

$$A(J)\ +\ jB(J)\ =\ \Big[X(J)+jY(J)\Big]*e^{j\ \left[\text{THT}-2\pi(\text{SIMF}\emptyset+\text{FI}*J)(\text{T}\emptyset+p\cdot\text{TJIT})\right]}$$

where p is a sample of a random process having a uniform distribution from -1/2 to 1/2.

CALL RSHFTS(X,Y,A,B)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
A contains the Output Waveform - R
B contains the Output Waveform - I

The basic mechanization equation for this entry point is the following:

$$A(J)+jB(J)=A(J)+jB(J)+\left[X(J)+jY(J)\right]*e^{j[THT-2\pi(SIMF\emptyset+FI*J)\cdot(T\emptyset+p\cdot TJIT)]}$$

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-63

b. Cross Reference Table: Page 9-216

6. THEORY OF OPERATION

If the Fourier transform pair is given for the signal s(t):

$$s(t) \longleftrightarrow S(f)$$

then $s(t-t_o) \longleftrightarrow S(f)\ e^{-j2\pi ft_o}$

This expression was derived in reference 1.

In addition the introduction of a constant phase shift in the time waveform results in the same constant phase shift in the frequency spectrum, e.g.

$$s(t)e^{j\emptyset} = S(f)\ e^{j\emptyset}$$

Note: Since these subroutines have classification code LTI-1 or 1S the input and output waveforms are in the frequency domain.

```
      SUBROUTINE SHIFT(X,Y,A,B)
      COMMON/BLK1/ BK1(200)
      EQUIVALENCE (BK1(180),TO),(BK1(181),THT),(BK1(182),TJIT)
     *            ,(BK1(8),SIMFO)
      DIMENSION X(1),Y(1),A(1),B(1)
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      DATA PI2,EX/6.2831853,2.9103830E-11/                             1780
C
      ICON=0
      TOR=TO
      GO TO 10
C
      ENTRY RSHIFT(X,Y,A,B)
```

```
CARD NO    ****                          CONTENTS

 1914            ICON=0
 1915            GO TO 27
 1916      C
 1917            ENTRY SHIFTS(X,Y,A,B)
 1918            ICON=1
 1919            TCR=T0
 1920            GO TO 10
 1921      C
 1922            ENTRY RSHFTS(X,Y,A,B)
 1923            ICON=1
 1924            GO TO 27
 1925      C
 1926         10 NPTS=IDOUL(X(N193))
 1927            IF(TCR.EQ.0.0)GO TO 20
 1928            NPTS2= NPTS/2
 1929            IPASS=1
 1930            K=NPTS2+1
 1931            KDEL=1
 1932            THT=THT/360.0
 1933            PH=-OMEGA*TCR+THT
 1934            DELPH=-X(N195)*TCR
 1935         15 PS= (PH-AINT(PH))*PI2
 1936            DELPS= (DELPH-AINT(DELPH))*PI2
 1937            ACDEL= COS(DELPS)
 1938            ASDEL= SIN(DELPS)
 1939            CS= COS(PS)
 1940            SN= SIN(PS)
 1941            GO TO 25
 1942      C
```

```
   22 THT=THT*PI/180.0
      ACCEL=1.0
      ASCEL=0.0
      CS= COS( THT)
      SN= SIN( THT)
      NPTS2=NPTS
      IPASS=2
      K=1
      KCEL=1
C
   25 IF(ICUR) 26,30,40
C
   27 RTAM=FLOAT(IFIX(RRAN(4)))*.X*(IJIT+IJIT)-IJIT
      TCNST=TC+RTAM.
      GO TO 30
C
   30 CONTINUE
      DO 300 L=1,NPTS2
      TIMF=X(K)
      X(K)=TIMF*CS-Y(K)*SN
      Y(K)=Y(K)*CS+TIMF*SN
      TEMP=CS
      CS=CS*ACCEL-SN*ASCEL
      SN=SN*ACCEL+TEMP*ASCEL
      K=K+KCEL
  300 CONTINUE
      GO TO 100
C
  100 CONTINUE
```

```
 CARD NO        ****                                 CONTENTS                                  ****

   1972                 DO 400 L=1,NPTS2
   1973                 TIMP=X(K)
   1974                 A(K)=A(K)+TIMP*CS-Y(K)*SN
   1975                 B(K)=B(K)+Y(K)*CS+TIMP*SN
   1976                 TEMP=CS
   1977                 CS=CS*ACDEL-SN*ASDEL
   1978                 SN=SN*ACDEL+TEMP*ASDEL
   1979                 K=K+KDEL
   1980             400 CONTINUE
   1981           C
   1982             500 IF(IPASS.NE.1) GO TO 600
   1983                 IPASS=2
   1984                 K=NPTS2
   1985                 KDEL=-1
   1986                 PPEFF=-DELFF
   1987                 DELPHA=-DELPH
   1988                 GO TO 25
   1989             600 CONTINUE
   1990           C
   1991                 A(N2P1)=X(N2P1)
   1992                 A(N2P4)=X(N2P4)
   1993                 A(N2P2)=X(N2P2)
   1994                 A(N2P5)=X(N2P5)
   1995                 [illegible](N2P4)=X(N2P4)
   1996                 [illegible](N2P3)=Y(N2P3)
   1997           C
   1998                 RETURN
   1999                 END
```

SUBROUTINE SWPINT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
SWPINT	LTI-4	434,435
NCSWPI	LTI-4	436,437

2. PURPOSE:

This subroutine is used to simulate a video post detection sweep integrator.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FBCK	R	F	Feedback coefficient

4. CALLING SEQUENCES:

CALL SWPINT (DIN, DOUT)

Where: DIN contains the Input Waveform

DOUT contains the Output Waveform

The range bin storage array (SRI) is cleared before execution begins.

CALL NCSWPI (DIN, DOUT)

Where: DIN contains the Input Waveform

DOUT contains the Output Waveform

The range bin storage array (SR1) is not cleared prior to execution except for the first execution of this subroutine.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The maximum number of range bins simulated is 2048. If more than 2048 range samples are contained in the input array, only the first 2048 are processed.

b. Flow Chart: Page 9-204

c. Cross Reference Table: Page 9-234

6. THEORY OF OPERATION

The block diagram of the video sweep integrator simulated by this module is shown in Figure SWPINT-1. The Z-plane transfer function for each range bin is given by the following expression:

$$T(Z) = \frac{Z}{Z - FBCK}$$

The delay represented by the Z operator is determined by the radar pulse repetition interval.

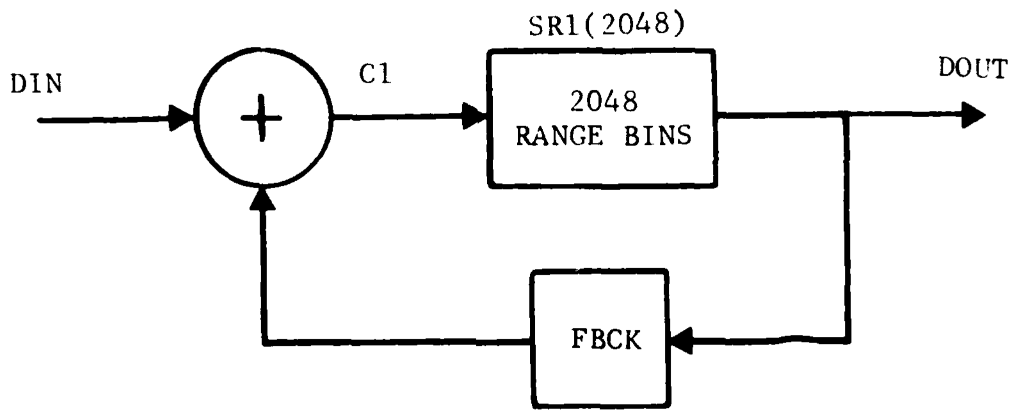


Figure SWPINT-1 BLOCK DIAGRAM OF SWPINT/NCSWPI

```
                                                                            UC7TS002
      SUBROUTINE SWPINT(DIN,DOUT)                                           UC7PL001
      COMMON/BLK1/BK1(500)                                                  UC7PL002
      DIMENSION DIN(1),DOUT(1),SK1(2048)                                    UC7PL003
      EQUIVALENCE (BK1( 21), IDMY        ), (BK1( 75), FLCK         )       UC7PL004
      DATA N193,N194,N195,N196/-3,-2,-1,0/                                  UC7PL005
      DATA IFLG/0/                                                                60
      ICON=1                                                                      61
      GO TO 40                                                                    62
      ENTRY RESWPI(DIN,DOUT)                                                      63
      ICON=2                                                                      64
   40 IF(IFLG.EQ.1.AND.ICON.EQ.2) GO TO 50                                        65
      DO 20 J=1,2048                                                        UC7PL007
      SK1(J)=0.0                                                            UC7PL008
   20 CONTINUE                                                              UC7PL009
      IFLG=1                                                                     100
   50 CONTINUE                                                                   101
      N = IFIX(DIN(N193))                                                   UC7PL011
      IF(N.LE.2048) GO TO 25                                                UC7PL012
      N=2048                                                                UC7PL013
      WRITE(6,35)                                                           UC7PL014
   35 FORMAT( ' TOO MANY POINTS IN INPUT ARRAY....FIRST 2048 PROCESSED')UC7PL015
   25 DOUT(N193)= FLOAT(N)                                                  UC7PL016
      DOUT(N194) = DIN(N194)                                                UC7PL017
      DOUT(N195) = DIN(N195)                                                UC7PL018
      DO 10 J=1,N                                                           UC7PL019
      D1=DIN(J)+SK1(J)*FLCK                                                 UC7PL020
      DOUT(J)=SK1(J)                                                        UC7PL021
      SK1(J)=D1                                                             UC7PL022
   10 CONTINUE                                                              UC7PL023
      RETURN                                                                UC7PL024
      END                                                                   UC7PL025
```

SUBROUTINE TARGET

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
TARGET	LTI-2	501
TGTNCL	LTI-2S	502

2. PURPOSE:

This subroutine simulates a target represented by a set of discrete scattering centers.

3. INPUT PARAMETERS:

Name	O/R	T	Description
FØ	R	F	RF Center Frequency
FEXT	R	F	Simulation bandwidth
FI	R	F	Frequency increment of the output transfer function
HTGT	O	F	Height of Target
RTGTØ	O	F	Range to target (R in Figure 1) (Used to compute target return variation as a function of range and altitude)
ANGTGT	O	F	Angle from the radar reference line to the target (θin Figure 1)
TOR1NT	O	F	Angle from Reference line #1 to the target coordinate system reference line (Ø in Figure 1). Reference line #1 is parallel to the radar coordinate system reference line and passes through the center of target rotation.

Name	O/R	T	Description
NSCAT	R	I	Number of scatterers
TSCAT	R	F	Array containing the location and radar cross section of each scatterer. TSCAT (1,k) is the radar cross section of the kth scatterer(σ_k in Figure 1). TSCAT(2,k) is the range coordinate of the kth scatterer in the target coordinate system (r_k in Figure 1). TSCAT(3,k) is the angle coordinate of the kth scatterer in the target coordinate system (β_k in Figure 1).
RØØØ	R	F	Range of target in the simulation (used to compute time delay only)
TGTVEL	0	F	Target radial velocity
TIME	0	F	Elapsed time since beginning of simulation

4. CALLING SEQUENCES:

CALL TARGET (X,Y)

Where: X contains the Output Waveform - R
Y contains the Output Waveform - I

CALL TGTNCL (X,Y)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The discrete scattering center representation used herein is obtained by imaging coherent short pulse measurements made on the target or a model of the target. The imaging procedure is described in Appendix H of this report.

b. The target representation implemented herein includes the pulse to pulse phase shift due to target motion with respect to the radar system. However, the intrapulse phase shift is not included in this model.

c. Flow Chart: Page 9-109

d. Cross Reference Table: Page 9-222

6. THEORY OF OPERATION

The discrete scatterer target scintillation model is based on the premise that a target can be represented as an ensemble of scattering centers. These scattering centers are assumed to be sufficiently small that they can be represented by impulses. Therefore, this model can be described by the following expression:

$$g(t) = \sum_{n=1}^{N} \sqrt{\sigma_n}\ \delta(t - t_n)$$

where σ_n *is the radar cross section of the nth scatterer*

t_n is the time delay of the nth scatterer

$\delta(\ )$ is the Dirac delta function.

The Fourier transform of the above expression is given by the following equation:

$$G(f) = \int_{-\infty}^{\infty} \sum_{n=1}^{N} \sqrt{\sigma}_n\ \delta(t - t_n)\ e^{-j2\pi ft}\ dt$$

$$G(f) = \sum_{n=1}^{N} \sqrt{\sigma}_n\ e^{-j2\pi ft_n}$$

In the target model the following equations were used in evaluating the target response in the frequency domain.

$$X(K) = G_{\epsilon}\ (ELANG) * G_{\psi}(ANGTGT) * \sum_{J=1}^{NSCAT} \sqrt{\sigma}_J \ COS\ 2\pi K(FI+FSTRT)t_n$$

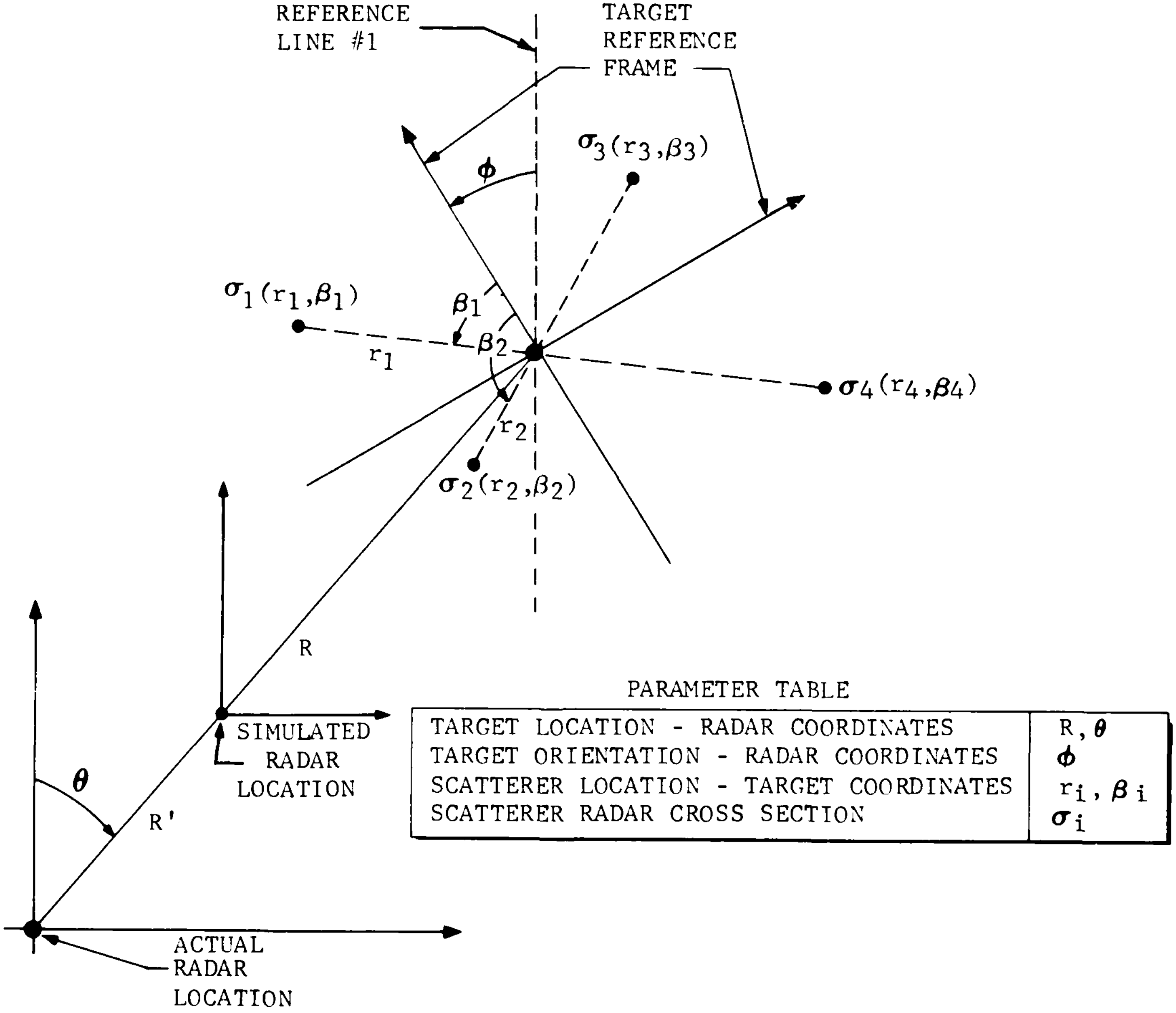
$$Y(K) = G_{\epsilon}(ELANG) * G_{\psi}(ANGTGT) * \sum_{J=1}^{NSCAT} \sqrt{\sigma}_J \ SIN\ 2\pi K(FI+FSTRT)t_n$$

where: K is the frequency index
FI is the frequency increment
FSTRT is the starting frequency
$FSTRT = FO - FEXT/2$
G_{ψ} is the antenna azimuth gain function
G_{ϵ} is the antenna elevation gain function

$ELANG = SIN^{-1}\ (HTGT/RTGT)$

t_n is the time displacement of the scatterer

$$t_n = \frac{1}{0.149896}\ RTGT + TSCAT(2,J)*COS(\theta-\emptyset-TSCAT(3,J))$$



PARAMETER TABLE

TARGET LOCATION - RADAR COORDINATES	R, θ
TARGET ORIENTATION - RADAR COORDINATES	ϕ
SCATTERER LOCATION - TARGET COORDINATES	r_i, β_i
SCATTERER RADAR CROSS SECTION	σ_i

Figure TARGET-1 Target Model Geometry

```
      SUBROUTINE TARGET(X,Y)
      COMMON/BLK1/ BK1(200),TSCAT(3,100)
      EQUIVALENCE    (  BK1(  3), FO         ),(  BK1(  4), FEXT       ),
     *               (  BK1( 11), FI         ),
     *               (  BK1( 21), IDMY       ),(  BK1(106), NTGT       ),
     *               (  BK1(107), KTGTO      ),(  BK1(108), ANGTGT     ),
     *               (  BK1(109), ICRINT     ),(  BK1(110), KODE       ),
     *               (  BK1(111), NSCAT      ),(  BK1(112), TGTVEL     )
     * ,             (  BK1( 16), TIME       )
      DIMENSION X(1) , Y(1)
       DATA N193,N194,N195,N196/-3,-2,-1,0/
      DATA PI2/6.2831853/
C
      ICON=1
      GO TO 10
C
      ENTRY TGTNEW(X,Y)
      ICON=0
C
   10 CONTINUE
      NFTS = IFIX(FEXT / FI )
      NFTS2= NFTS/2
      NFTS=  NFTS2*2
      IF(ICON.EQ.0) GO TO 40
C
      DO 20 J=1,NFTS
      X(J)=0.
      Y(J)=0.
```

```
CARD NO     ****                    CONTENTS                    ****

             30 CONTINUE
            C
             40 CONTINUE
                A1=ANG(J)-FANG
                GE=1.0
                FREF=FCOO+TGTVEL*TIME
                FTGT=FTGTO+TGTVEL*TIME
                IF(HTGT.GT.0.0) GE=ELGAIN(ARSIN(HTGT/FTGT)*57.2957)
             66 G=AZGAIN(ANGLET)*GE
            C
                DO 60 J=1,NSCAT
                FI=FREF + TSCAT(2,J)*COS((A1-TSCAT(1,J))*0.01745329)
                T=FI/0.[illegible]
                A=G*SQRT(TSCAT(3,J))*G
                DELTA=T-T1
                PHASE=FC*T
                IT=T/DT
                K=IT/SC+1
                KOL=1
             60 CONTINUE
                DELPS=AMOD(DELPH,1.0)*PI2
                XDELPS=COS(DELPS)
                YDELPS=SIN(DELPS)
                PHASE=AMOD(PHASE,1.0)*PI2
                IS=A*COS(PHASE)
                QS=A*SIN(PHASE)
            C
                DO 70 L=1,NPTS
                XI(L)= IS*C
```

```
      Y(K)=Y(K)+SN
      TEMP=CS
      CS=CS*ACDEL-SN*ASDEL
      SN=SN*ACDEL+TEMP*ASDEL
      K=K+KDEL
  400 CONTINUE
C
      IF(IPASS.NE.1) GO TO 200
      IPASS=2
      K=NPTS2
      KDEL=-1
      DPHAS=DPHAS-DELPH
      DELPH=-DELPH
      GO TO 71
  200 CONTINUE
C
      X(N193) = ALOG(NPTS)
      X(N194)=-PI*FLOAT(NPTS/2)
      X(N195) = PI
      Y(N193) = X(N193)
      Y(N194) = X(N194)
      Y(N195) = X(N195)
C
C
      RETURN
      END
```

SUBROUTINE WEITRE

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
WEITRE	LTI-1 or LTV-4	221
WEITCP	LTI-1 or LTV-4	222
WEITMP	LTI-1 or LTV-4	223

2. PURPOSE:

This subroutine is used to represent (1) a linear time variant device in terms of its measured transfer functions or (2) a linear time varying device in terms of its measured gain (complex) as a function of time.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NPWT	R	I	Number of points used to specify the weighting function
WT	R	F	Array containing the specified weighting function. The specification for the Jth sample is the following: WEITRE or WEITCP WT (1,J) = Real component WT (2,J) = Time WT (3,J) = Imaginary component (WEITCP only) WEITMP WT (1,J) = Gain WT (2,J) = Time

			WT (3,J) = Phase angle
ORIG	0	F	Amount the independent variables of the weighting function is to be shifted.

4. CALLING SEQUENCES:

Real Weighting Function

CALL WEITRE (X, Y, $mmmm)

Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

Complex Rectangular Weighting Function
CALL WEITCP (X, Y, $mmmm)
Where: X contains the Input Waveform - R
Y contains the Input Waveform - I
X contains the Output Waveform - R
Y contains the Output Waveform - I

Complex Polar Weighting Function
CALL WEITMP (X, Y, $mmmm)
Where: X contains the Input Waveform - M
Y contains the Input Waveform - P
X contains the Output Waveform - M
Y contains the Output Waveform - P

$mmmm is the number of the statement to which control is transferred if an error is detected. in the input data.

5. RESTRICTIONS, REQUIREMENTS, AND MISCELLANEOUS DATA

a. The input weighting function independent variable must be monotonically increasing, except that two adjacent points may be specified with the same value. In this case, if the input value to be weighted falls exactly on that point, the first dependent value specified will be used.

b. If the extent of the weighting function does not include any part of the input data, the subroutine will perform a nonstandard exit.

c. If the extent of the weighting function contains only a portion of the input data, all other input points are set to zero.

d. The subroutine performs a linear interpolation between specified points.

e. Flow Chart: Page 9-57

f. Cross Reference Table: Page 9-215

6. THEORY OF OPERATION

The basic mechanization equations for each entry point are as follows:

WEITCP:

```
X(J) = X(J)*XWT - Y(J) * YWT
Y(J) = X(J)*YWT + Y(J) * XWT
```

A table lookup/linear interpolation scheme is used to determine the complex weight, XWT + j YWT. The value of the independent variable used in computing the weight for the Jth sample is given by the following expression:

$$VIV = XORIG + J \cdot DEL - ORIG$$

where: XORIG = Independent variable for first element of the input arrays.

DEL = Independent variable increment between samples of the input array.

WEITRE:

```
X(J) = X(J) * XWT

Y(J) = Y(J) * XWT
```

The procedure for determination of the weight, XWT, is the same as that described above except YWT is not computed.

WEITMP:

```
X(J) = X(J) * AMPL

Y(J) = Y(J) + PHAS
```

A table lookup/interpolation scheme is used to determine the complex weight AMPL * EXP (PHAS). The procedure for determining value of the independent variable used in computing the weight is described under WEITCP.

```
      SUBROUTINE WEITRC ( X, Y,*)
C
C**** THIS SUBROUTINE PERFORMS A WEIGHTING OPERATION ON THE INPUT ARRAYS
C     X AND Y, USING THE WEIGHTING ARRAY, WT.  THE WEIGHTING OPERATION MAY
C     BE PERFORMED AS REAL ON COMPLEX (WEITRC), COMPLEX ON COMPLEX (WEITCP)
C     OR MAGNITUDE AND PHASE ON MAGNITUDE AND PHASE (WEITMP). *************
C
      COMMON/BLNK1/BK1(500)
      DIMENSION X(1), Y(1), WT(3,100)                                   WHT04
     EQUIVALENCE  (BK1( 1)),IDMY        ),
     .              (BK1( 37), NEWT        ), (BK1( 38), CRIO        ),
     .              (BK1(201), WT(1,1)     )
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      DATA A1/Z4505C44E05/                                               460
      ICON = 0
      GO TO 5
      ENTRY WEITCP ( X, Y, *)
      ICON = 1
      GO TO 5
      ENTRY WEITMP (X, Y, *)                                             W
      ICON = 2
    5 TI = X(N193)
      T = X(N194) - CRIO
      NF = IFIX(X(N195))                                                 W
      TEXT = T + TI*FLOAT(NF)
      IF(NEWT.LT.2 .OR. (TEXT.LT.WT(2,1) .OR. T.GT.WT(2,NEWT))) GO TO 500
      IF(T.GE.WT(2,1) .AND. TEXT.LE.WT(2,NEWT)) GO TO 400
      WRITE (6,1001)
  400 CONTINUE
        T=WT(2,2)-WT(2,1)
       IF(ICON.EQ.0) GO TO 200
       SLOPE1=(WT(1,2)-WT(1,1))/DT
        SLOPE3=(WT(3,2)-WT(3,1))/DT
        GO TO 201
  200  SLOPE1=0.0
       SLOPE3=0.0
       K=1
       [illegible]
       [illegible]
       [illegible]
          [illegible]
```

```
CARD NO.   ****                              CONTENTS                              ****

           220 K=K+1
               DT = WT(2,K+1) - WT(2,K)
               IF (DT .EQ. 0.0) GO TO 220
               SLOPE1 =(WT(1,K+1)-WT(1,K)) /DT                                     WH
               SLOPE3 = (WT(3,K+1)-WT(3,K)) /DT                                    WH
           250 IF (K .GE. NPWT) GO TO 275
               TIMX = T - WT(2,K)
               IF ( TIMX ) 275, 270, 280
           270 XT= WT(1,K)
               IF (ICON .EQ. 1 .OR. ICON .EQ. 3) YT = WT(3,K)
               GO TO 275
           280 IF(T.GE.WT(2,K+1)) GO TO 220
               XT = TIMX * SLOPE1 + WT(1,K)
               IF(ICON.EQ.1.OR.ICON.EQ.3) YT=TIMX*SLOPE3+WT(3,K)
           275 CONTINUE
               IF (ICON .GT. 1) GO TO 285
               A = X(J)
               X(J) = A * XT - Y(J) * YT
               Y(J) = A * YT + Y(J) * XT
               GO TO 290
           285 X(J) = X(J)*XT
               Y(J)=Y(J)+YT
           290 T = T +TI
           300 CONTINUE
               RETURN
           900 WRITE (6,1000)
               RETURN 1
          1000 FORMAT(' THE WEIGHTING ARRAY, WT, IS NOT DEFINED OVER THE COMPLETE
              1INPUT DATA RECORD' THE OUTPUT IS SET TO ZERO AT THE UNDEFINED POI
              2NTS.' / / )
          1010 FORMAT(1H ,'THE WEIGHTING ARRAY, WT, IS NOT PROPERLY DEFINED.  TH
              1E WEIGHTING OPERATION WILL NOT BE PERFORMED.'// )
```

SUBROUTINE WVGU1D

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
WVGU1D	LTI-1	510

2. PURPOSE:

This routine is used to simulate a length of waveguide for the purpose of predicting the effect of nonlinear phase behavior on waveforms.

3. INPUT PARAMETERS:

Name	O/R	T	Description
RFF	R	F	Center frequency of the electromagnetic wave propagating in the waveguide
CFREQ	R	F	Cutoff frequency of the waveguide
XWLENG	R	F	Length of the waveguide

4. CALLING SEQUENCES:

CALL WVGUID (X,Y)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. References:

Atwater, H. A.; Introduction to Microwave Theory, McGraw-Hill, New York, 162, Chapter 3.

b. Flow Chart: Page 9-116

c. Cross Reference Table: Page 9-223

6. THEORY OF OPERATION

A hollow rectangular perfectly conducting waveguide excited in only one of its modès has a propagation function which is described by the following expression:

$$\beta_g(f) = \sqrt{k^2 - k_c{}^2}$$

where: $k = 2\pi f\sqrt{\mu_o \epsilon_o}$

$k_c = 2\pi f_c\sqrt{\mu_o \epsilon_o}$

f_c = cutoff frequency

Substituting $\frac{1}{C^2} = \mu_o \epsilon_o$ the following expression is obtained:

$$\beta_g(f) = \frac{2\pi}{C}\sqrt{f^2 - f_c{}^2}$$

Therefore the transfer function for a length of waveguide considering only the phase term is given by the following expressions:

$$G(f) = e^{-j\frac{2\pi}{C} z \sqrt{f^2 - f_c{}^2}} \quad ; f_c < f$$

$$= e^{\frac{2\pi}{C} z \sqrt{f_c{}^2 - f^2}} \quad ; f \leq f_c$$

where z is the length of the waveguide.

```
CARD NO      ****                    CONTENTS                                    ****

  3074               SUBROUTINE WVGUID(X,Y)
  3075               COMMON/BLK1/ VAR(500)
  3076               DIMENSION X(1),Y(1)
  3077               EQUIVALENCE (VAR(146) , CFREQ ) , (VAR(147) , XWLENG )
  3078               EQUIVALENCE (VAR(  3) , FREQ )
  3079               DATA N193,N194,N195/-1,-2,-1/,CNST/3.33357/,PI2/6.283185/
  3080         *  UNITS:  CFREQ(GHZ) , XWLENG(METERS) , CNST(NANOSEC/METER)
  3081               CF2=CFREQ*CFREQ
  3082               T1=CNST*XWLENG
  3083               T2= T1*PI2
  3084               NPTS=DBLE(X(N193))
  3085               FREQ=X(N194)+FREQ
  3086               DELF=X(N195)
  3087               DO 200 J=1,NPTS
  3088               F2=FREQ*FREQ
  3089               IF(F2.LE.CF2) GOTO 100
  3090               THETA=T1*SQRT(F2-CF2)
  3091               THETA= (THETA-FLOAT(IFIX(THETA)))* PI2
  3092               C=COS(THETA)
  3093               S=-SIN(THETA)
  3094               GOTO 150
  3095         100    C=EXP(-T2*SQRT(CF2-F2))
  3096               S=0.0
  3097         150    TMP=X(J)
  3098               X(J)=X(J)*C-Y(J)*S
  3099               Y(J)=TMP*S+Y(J)*C
  3100               FREQ=FREQ+DELF
  3101         200  CONTINUE
  3102               RETURN
```

SECTION 5

CONNECTION MODULES

This section includes all those modules that bind the stimulus/transfer function modules together in order to form the core of the simulation. The functions performed by these modules include array addition or array multiplication (CONV) and conversion between time and frequency domain representations (ZFFT).

SUBROUTINE CONV

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CONV	Connection	204
CONVMP	Connection	205
DIVA	Connection	206
ADDA	Connection	207,239

2. PURPOSE:

This subroutine serves to connect simulation modules together by performing array addition, multiplication and division.

3. INPUT PARAMETERS:

None

4. CALLING SEQUENCES:

Complex Rectangular Array Multiplication

```
CALL  CONV (XI, XQ, FI, FQ, $mmmm)
```

```
Where:     XI contains the Input Waveform - R   Complex
           XQ contains the Input Waveform - I   Array #1
           FI contains the Input Waveform - R   Complex
           FQ contains the Input Waveform - I   Array #2
           XI contains the Output Waveform - R
           XQ contains the Output Waveform - I
```

mmmm is the return address for a non-standard return

$$XI(J) = XI(J) * FI(J) - XQ(J) * FQ(J)$$
$$XQ(J) = XI(J) * FQ(J) + XQ(J) * FI(J)$$

Complex Polar Array Multiplication

CALL CONVMP (XI, XQ, FI, FQ, $mmmm)

Where:
XI contains the Input Waveform - M Complex
XQ contains the Input Waveform - P Array #1
FI contains the Input Waveform - M Complex
FQ contains the Input Waveform - P Array #2
XI contains the Output Waveform - M
XQ contains the Output Waveform - P

mmmm is the return address for a non-standard return.

XI(J) = XI(J) * FI(J)
XQ(J) = XQ(J) + FQ(J)

Complex Polar Array Division

CALL DIVA (XI, XQ, FI, FQ, $mmmm)

Where:
XI contains the Input Waveform - M Complex
XQ contains the Input Waveform - P Array #1
FI contains the Input Waveform - M Complex
FQ contains the Input Waveform - P Array #2
XI contains the Output Waveform - M
XQ contains the Output Waveform - P

mmmm is the return address for a non-standard return

XI(J) = XI(J)/FI(J)
XQ(J) = XQ(J)-FQ(J)

Array Addition

CALL ADDA (XI, FI, $mmmm)

Where:
XI contains the Input Waveform
FI contains the Input Waveform
XI contains the Output Waveform

mmmm is the return address for a non-standard return

XI(J) = XI(J) + FI(J)

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The difference in the independent variable increment of the input arrays must be less than or equal to 10^{-4}.

b. The output is set to zero for any cases where the independent variables of the two arrays do not overlap.

c. When polar operations are performed the phase angle is measured in degrees.

d. If the arrays do not overlap, or if they must be shifted more than 4097 increments, the problem will abort and the non-standard return will be used.

e. Flow Chart: Page 9-60

f. Cross Reference Table: Page 9-216.

6. THEORY OF OPERATION

Depending upon the entry point through which the subroutine is called, a control flag is set to an integer number 1 to 4. This number indicates to the subroutine the type of operation which is to be performed, either rectangular multiplication, polar multiplication, polar division, or addition. After the flag is set, the two arrays are examined (beginning at statement number 1000) to verify that the values of the increments of the two arrays are within 1/10000 of each other. The arrays are inspected to verify that they overlap, at least in part, and one array is shifted, if required, to align the two origins. If the increments are not within 1/10000 of each other, or if the arrays do not overlap, or if the origins of the two arrays are not within 4097 increments of each other, the problem is aborted. Otherwise, the proper arithmetic manipulation is performed, according to the value of the control flag and the results are placed in the X array(s). The F array(s) remain unmodified after exit from the subroutine. For the divide operation, the X complex array is the dividend and the F complex array is the divisor.

```
      [illegible] X(I),[illegible]I,[illegible],*)
C
      [illegible] Z(I),X(I),F(I),[illegible]
      [illegible] N194,N195,N196 [illegible]
C
      [illegible]
      [illegible]
      [illegible] X,[illegible],*)
      [illegible]
      [illegible]
      [illegible] (X,X,I),[illegible],*)
      [illegible]
         [illegible]
      [illegible] (X,I),*)
      [illegible]
      [illegible]
      [illegible] (XI(N195)-FI(N195)) * [illegible]
      IF ([illegible] XI(N195)) .LT. 200
      [illegible] = AMAX1 (XI(N194) , FI(N194))
      [illegible] (XI(N195))
      [illegible] (FI(N195))
      [illegible]=XI([illegible]) + XI(N195) * FLOAT(NX1)
      [illegible] FI([illegible]) + FI(N195) * FLOAT(NF1)
      [illegible] AMIN1([illegible])
      [illegible] IFIX ([illegible]ENDATA-[illegible](N) / XI(N195) + 0.50)
```

```
 CARD NO.      ****                        CONTENTS                                  ****

  1656                  IXI=1+IFIX( (ORIGN-XI(N194))/ XI(N195) + 0.40)
  1657                  IFI=1+IFIX( (ORIGN-FI(N194))/ XI(N195) + 0.40)
  1658                  IF(IXI+NPTS.GT.NXI+1) NPTS=NPTS-1
  1659                  IF(IFI+NPTS.GT.NFI+1) NPTS=NPTS-1
  1660                  WRITE(6,50) NPTS,IXI,IFI,NXI,NFI,ORIGN,ENDATA,XI(N194),FI(N194)
  1661               50 FORMAT(1H ,5I12,4E14.6)
  1662                  XI(N193)=BOOL(NPTS)
  1663                  XI(N194)=ORIGN
  1664                  IF(ICON.EQ.4) GO TO 700
  1665              701 XC(N193) = XI(N193)
  1666                  XC(N194) = XI(N194)
  1667              700 CONTINUE
  1668           C************************************************************************
  1669                  DO 20 J=1,NPTS
  1670                  GO TO (101,102,103,104),ICON
  1671           C
  1672              101 CONTINUE
  1673                  A=XI(IXI)
  1674                  XI(J)= A*FI(IFI) - XC(IXI)*FC(IFI)
  1675                  XC(J)= A*FC(IFI) + XC(IXI)*FI(IFI)
  1676                  GO TO 20
  1677           C
  1678              102 CONTINUE
  1679                  XI(J)= XI(IXI)*FI(IFI)
  1680                  XC(J)= XC(IXI)+FC(IFI)
  1681                  GO TO 20
  1682           C
  1683              103 CONTINUE
  1684                  XI(J)= XI(IXI)/FI(IFI)
```

```
          XE(J)= XE(IXI)-FE(IFI)
          GO TO 25
C
  104 CONTINUE
          XI(J)= XI(IXI)+FI(IFI)
C
   25 IXI=IXI+1
          IFI=IFI+1
   20 CONTINUE
          WRITE(6,50) IXI,IFI
C
          RETURN
  200 WRITE(6,201)
  201 FORMAT(' INDEPENDENT VARIABLE INCREMENTS DO NOT MATCH ')
  202 RETURN 1
          END
```

SUBROUTINE ZFFT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
ZFFT	Connection	202
ZIFFT	Connection	203
IFT	Connection	Not User Referenced

2. PURPOSE:

This subroutine performs the finite discrete Fourier Transform (ZFFT) and inverse Fourier Transform (ZIFFT or IFT) of a sequence of input data samples. The mechanization is based on the Fast Fourier Transform (FFT) algorithm developed by Langdon and Sande from the approach of J. W. Tukey and J. Cooley.

3. INPUT PARAMETERS:

Name	O/R	T	Description
N2	R	I	Power of 2 which determines the total number of prints (NTHPOW) transformed by the FFT.
FSHIFT	R	F	Frequency shift to be applied to frequency domain representation of the waveform.
ICPLXI	O	I	Control flag which indicates the nature of the inverse transform (ZIFFT) input data = 1 complex waveform Z = X + jY = 0 real waveform Z = X
ICPLXF	O	I	Control flag which indicates the nature of the forward transform (ZFFT) input data = 1 complex waveform Z = X + jY = 0 real waveform Z = X
SIMBW	R	F	Width of output spectrum when forward transform (ZFFT) is performed

Name	O/R	T	Description
INORM	R	I	Normalization flag

	ZFFT	ZIFFT
= 0;	$f_N = TI$	$f_N = FI$
= 1;	$f_N = 1.0/NP$	$f_N = FI$
= 2;	$f_N = TI$	$f_N = 1.0/(2**N2)$
= 3;	$f_N = 1.0/NP$	$f_N = 1.0/(2**N2)$
= 4;	$f_N = 1.0$	$f_N = 1.0$

4. CALLING SEQUENCES:

Fourier Transform

CALL ZFFT (X,Y)

Where:

X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

Increase Fourier Transform

CALL ZIFFT (X, Y)

Where:

X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

Inverse FFT only

CALL IFT (X, Y)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

X contains the Output Waveform - R

Y contains the Output Waveform - I

This entry point is used in computing antenna patterns and is called by ANTPAT module only.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The maximum value of N2 is 13 which gives 8192 transformed samples.

b. The number of points transformed by the FFT algorithm is 2 raised to an integer power. If the number of input samples is not equal to a power of 2, then zeros are used to fill in the array elements not defined by the input data; i.e., zeros are placed in the input array locations from NP + 1 to 2**N2.

c. When the Forward Transform is executed the output spectrum is shifted by FSHFT and has a width of SIMBW: i.e., the output spectrum extends from FSHIFT - SIMBW/2 to FSHIFT + SIMBW/2.

d. When the Forward Transform is executed the origin of the independent variable X (N194) is set to the discrete frequency nearest FSHIFT - SIMBW/2 since the FFT output consists of a discrete set of frequency samples. When the Inverse Transform is executed the origin of the independent variable X(N194) is set to zero.

e. The independent variable increment is set to 1.0/(X(N195)*(2**N2POW)) for both the forward and inverse transform.

f. When the inverse transform is executed the input spectrum is shifted by FSHIFT before the FFT is executed. The value of FSHIFT when the inverse transform is performed can be different from the value used when the forward transform was performed.

The following relationship must be satisfied

$$- {}^{FS}/_{2} \le \text{FSHIFT} \pm \text{SIMBW}/_{2} < {}^{FS}/_{2}$$

g. Flow Chart: Page 9-125

h. Cross Reference Table: 9-224

6. THEORY OF OPERATION

a. ZFFT - This entry point connects the time domain representation of a waveform into the corresponding frequency domain representation. The Fourier transform relationship is given by the following expression:

$$S(f) = \int_{-\infty}^{\infty} s(t)\, e^{-j2\pi ft}\, dt$$

The basic mechanization equation used in computing the discrete Fourier transform is the following:

$$S_k = f_N \sum_{0}^{NP-1} s_i \quad W^{-k_i}$$

where: k is the frequency index 0 k NTHPOW-
i is the time index 0 i NP -1
f_N is the normalization factor
$W = \exp(j2\pi/\text{NTHPOW})$

In general s_i and S_k are complex arrays.

b. ZIFFT and IFT - These entry points connect the frequency domain representation of a waveform into the corresponding time domain representation. The inverse Fourier transform relationship is given by the following expression:

$$s(t) = \int_{-\infty}^{\infty} S(f)\, e^{j2\pi ft}\, dt$$

The basic mechanization equation used in computing the inverse discrete Fourier transform is the following:

$$s_i = f_N \sum_{0}^{NTHPOW-1} S_k W^{ki} \qquad 0 \leq i \leq NTHPOW-1$$

The only difference between ZIFFT and IFT is that IFT performs the inverse FFT only and no normalization is performed. The FFT mechanization utilized in this module evaluates the above expression directly. In order to compute the forward transform using the same inverse FFT the following expression is used.

$$S_k = f_N \left[\sum_{0}^{NP-1} s_i^* W^{ki} \right]^*$$

5-12

```
      SUBROUTINE ZFFT(X,Y)
      COMMON/ZERK1/ EK1(500)
      DOUBLE PRECISION CEEL,SEEL,CS,SN,CT,PI2,SCALE                     31
      EQUIVALENCE  ( N2  ,EK1(1))  ,(ISHIFT,EK1(5)) ,(IOFLX1,EK1(6)),
     *             (ICPLXF,EK1(7)) ,(SIMEW,EK1(8)) ,(IRESN ,EK1(9))
      DATA PI2/6.2831853066706/                                         61
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      REAL X(1),Y(1),I,I1,I2,I3,I4
      INTEGER PASS, SEGLOC, L(14)
      EQUIVALENCE (J,J1), (N4POW,J5), (PASS,J6),
     1    (NXTLTH,J7), (LENGTH,J8), (SEGLOC,J9), (ISCALE,J10),
     2    (IARG,J11) , (A1,J12) , (C2,L1), (C3,L2), (S1,L3),
     3    (S2,L4),(S3,L5),(R1,L6),(R2,L7),(R3,L8),(R4,L9),
     4 (I,I1)
      EQUIVALENCE (L14,L(1)),(L13,L(2)),(L12,L(3)),(L11,L(4)),
     1    (L10,L(5)),(L9,L(6)),(L8,L(7)),(L7,L(8)),(L6,L(9)),(L5,L(10)),
     2    (L4,L(11)),(L3,L(12)),(L2,L(13)),(L1,L(14))
      IFLAG = 1
      GO TO 5
      ENTRY ZIFFT(X,Y)
      IFLAG=-1
      GO TO 5
      ENTRY ZFFT( ,Y)
      IFLAG=
    9 CONTINUE
      NTHPOW = 2** N2
      NTTL2=NTHPOW/2
      NP= BOOL(X(N193))
      GO TO(502,501,411),IFLAG
  502 R1=X(N195)
       R2=1.0/R1
       IF(SIMBW.GT.R2) SIMBW=R2
      IF(INORM.EQ.1.OR.INORM.EQ.3) R1=1.0/FLOAT(NP)
      IF(INORM.EQ.4) R1=1.0
      IF(ICPLXF.EQ.0) GO TO 700
      DO 11 J=1,NP
      X(J)= X(J)*R1
      Y(J)=-Y(J)*R1
   11 CONTINUE
      GO TO 701
  700 DO 702 J=1,NP
      X(J)= X(J)*R1
      Y(J) =0.0
  702 CONTINUE
  701 CONTINUE
       IF(NTHPOW-NP) 32,31,30
   30  L1=NP+1
      DO 10 J=L1,NTHPOW
      X(J) = 0.0
      Y(J) = 0.0
   10 CONTINUE
       GOTO 500
   32  WRITE(6,33) NP,NTHPOW
```

```
CARD NO       ****                          CONTENTS                          ****

  3306        53    FORMAT(' NUMBER OF INPUT SAMPLES',I4,'EXCEEDS SPECIFIED TIME',
  3307            * ' SPAN=2**N2=',I4)
  3308      [illegible] CONTINUE
  3309              GO TO 500
  3310        501 CONTINUE
  3311              IF (NF.LE.NTHFOW.OR.N2.EQ.13) GO TO 399
  3312              N2=N2+1
  3313              WRITE (6,310) N2
  3314        310 FORMAT (' THE SIZE OF THE TRANSFORM ARRAY HAS BEEN EXPANDED',
  3315            * 'TO 2**',I2)
  3316              GO TO 9
  3317        399 R1= X(N194)+FSHIFT
  3318              R2=-FLOAT(NITTL2)*X(N195)
  3319              NSHIFT= IFIX( (R1-R2) / X(N195) + 0.40 )
  3320              B1=  R1 - X(N195)/2.0
  3321              T1=  R1 + X(N195)/2.0
  3322         59 R3=R2+ X(N195)* FLOAT( NSHIFT)
  3323              WRITE(6,600) NSHIFT,NF,R1,R2,R3,B1,T1,X(N194)
  3324        600 FORMAT(1H ,2I10,6E15.6)
  3325              IF(R3.GE.B1.AND.R3.LE.T1) GO TO 60
  3326              IF(R3.LT.B1) NSHIFT=NSHIFT+1
  3327              IF(R3.GT.T1) NSHIFT=NSHIFT-1
  3328              GO TO 59
  3329         60 IF(NSHIFT.LT.0) GO TO 404
  3330              L2=NF+NSHIFT
  3331              L3=NF
  3332              IF(L2.LE.NTHFOW) GO TO 370
  3333              IF(N2.EQ.13) GO TO 380
  3334              N2=N2+1
```

```
       WRITE(6,310) N2
       GO TO 9
  380   WRITE(6,385) L2,NTHPOW
  385 FORMAT(' THE NUMBER OF ARRAY ELEMENTS REQUIRED AFTER HETERODYNING',
      *I5, 'EXCEEDED AVAILABLE STORAGE...ARRAY REDUCED TO',I5,
      *' BY DELETING HIGH FREQ TERMS')
       L3=NTHPOW-NSHIFT
       L2=NTHPOW
  370   IF(NSHIFT.EQ.0) GOTO 430
       DO 400 L1=1,NF
       X(L2)=  X(L3)
       Y(L2)=  Y(L3)
       L2=L2-1
       L3=L3-1
  400  CONTINUE
       DO 401 L1=1,NSHIFT
       X(L1)=0.0
       Y(L1)=0.0
  401  CONTINUE
  430  L2=NF+NSHIFT+1
        IF(L2.GT.NTHPOW) GOTO 403
       DO 402 L1=L2,NTHPOW
       X(L1)=0.0
       Y(L1)=0.0
  402  CONTINUE
       GO TO 403
  404   IF(N2.EQ.13) GOTO 407
        N2=N2+1
        WRITE(6,310) N2
```

```
CARD NO        ****                            CONTENTS                                    ****

  3364                 GOTO 9
  3365          407    L3=NP+NSHIFT
  3366                 DO 405 L1=1,L3
  3367                 X(L1)=X(L1-NSHIFT)
  3368                 Y(L1)=Y(L1-NSHIFT)
  3369          405    CONTINUE
  3370                 L3=L3+1
  3371                 DO 406 L1=L3,NTHPOW
  3372                 X(L1)=0.0
  3373                 Y(L1)=0.0
  3374          406    CONTINUE
  3375                 L1=IABS(NSHIFT)
  3376                 WRITE(6,408) L1
  3377          408    FORMAT(' NUMBER OF LOCATIONS FREED AFTER HETERODYNING',
  3378               * 'EXCEEDED AVAILABLE STORAGE...',I5,' ELEMENTS DELETED',
  3379               * 'FROM NEGATIVE END OF ARRAY')
  3380           409 CONTINUE
  3381              IF(ICPLX1.EQ.1) GO TO 410
  3382              L2=NTHPOW+2
  3383              DO 420 J=2,NTHL2
  3384              L3=L2-J
  3385              X(J)=0.5*(X(J)+X(L3))
  3386              X(L3)=X(J)
  3387              Y(J)=0.5*(Y(J)-Y(L3))
  3388              Y(L3)=-Y(J)
  3389           420 CONTINUE
  3390           410 CONTINUE
  3391               R3=X(NP95)
  3392              IF(INOFFT.EQ.0.OR.NORM.EQ.3) R3=1.0/FLOAT(NTHPOW)
```

5-17

```
      IF(INORM.EQ.4) R3=1.0
      L1=NITL2
      DO 409 J=1,NITL2
      L1=L1+1
      R1=X(L1)*R3
      R2=Y(L1)*R3
      X(L1)=X(J)*R3
      Y(L1)=Y(J)*R3
      X(J)=R1
      Y(J)=R2
  409 CONTINUE
  411 CONTINUE                                                  1050
      X(N193)=LLUL(NTHPOW)
      X(N194)=0.0
  500 CONTINUE
      X(N195)=1.0/(FLOAT(NTHPOW)*X(N195))
      N4POW =  N2  /2
      IF(N4POW.EQ.0) GO TO 3
      DO 2 PASS=1,N4POW                                         1120
      NXTLTH=2**( N2  -2*PASS)
      LENGTH=4*NXTLTH
      SCALE=PI2/DBLE(FLOAT(LENGTH))                             1160
      CDEL=DCOS(SCALE)                                          1151
      SDEL=DSIN(SCALE)                                          1152
      CS=1.0D0                                                  1153
      SN=0.0D0                                                  1154
      DO 2 J=1,NXTLTH
      C1=CS                                                     1180
      S1=SN                                                     1190
```

```
 CARD NO       ****.                          CONTENTS                             ****

   3422            C2=CS*CS-SN*SN                                                  1200
   3423            S2=2.000*CS*SN                                                  1210
   3424            C3=C1*C2-S1*S2                                                  1220
   3425            S3=C2*S1+S2*C1                                                  1230
   3426            C1=CS                                                           1240
   3427            CS=CS*CDEL-SN*SDEL                                              1250
   3428            SN=C1*SDEL+SN*CDEL                                              1260
   3429            DO 2 SELECC=LENGTH,NTHPOW,LENGTH                                1410
   3430            J1 = SELECC-LENGTH+J
   3431            J2 = J1+NXTLTH
   3432            J3 = J2+NXTLTH
   3433            J4 = J3+NXTLTH
   3434            R1 = X(J1)+X(J3)
   3435            R2 = X(J1)-X(J3)
   3436            R3 = X(J2)+X(J4)
   3437            R4 = X(J2)-X(J4)
   3438            I1 = Y(J1)+Y(J3)
   3439            I2 = Y(J1)-Y(J3)
   3440            I3 = Y(J2)+Y(J4)
   3441            I4 = Y(J2)-Y(J4)
   3442            X(J1) = R1+R3
   3443            Y(J1) = I1+I3
   3444            X(J3)= C1*(R2-I4)-S1*(I2+R4)
   3445            Y(J3)= S1*(R2-I4)+C1*(I2+R4)
   3446            X(J2) = C2*(R1-R3) - S2*(I1-I3)
   3447            Y(J2) = S2*(R1-R3) + C2*(I1-I3)
   3448            X(J4) = C3*(R2+I4) - S3*(I2-R4)
   3449            Y(J4) = S3*(R2+I4) + C3*(I2-R4)
   3450          2 CONTINUE
```

```
    3 IF( N2  .EQ.2*N4POW) GO TO 5
      DO 4 J=1,NTHPOW,2
      R=X(J)+X(J+1)
      X(J+1)=X(J)-X(J+1)
      X(J) = R
      I=Y(J)+Y(J+1)
      Y(J+1)=Y(J)-Y(J+1)
    4 Y(J)=I
    5 DO 6 J=1,14
      L(J)=1
    6 IF(J.LE. N2  ) L(J)=2**( N2  +1-J)
      IJ=1
      DO 7 J1=1,L1
      DO 7 J2=J1,L2,L1
      DO 7 J3=J2,L3,L2
      DO 7 J4=J3,L4,L3
      DO 7 J5=J4,L5,L4
      DO 7 J6=J5,L6,L5
      DO 7 J7=J6,L7,L6
      DO 7 J8=J7,L8,L7
      DO 7 J9=J8,L9,L8
      DO 7 J10=J9,L10,L9
      DO 7 J11=J10,L11,L10
      DO 7 J12=J11,L12,L11
      DO 7 J13=J12,L13,L12
      DO 7 JI =J13,L14,L13
      IF(IJ.GE.JI) GO TO 7
      R=X(IJ)
      X(IJ)=X(JI)
```

```
 CARD NO        ****                                CONTENTS                                  ****

   3480               X(JI)=R
   3481               I=Y(IJ)
   3482               Y(IJ)=Y(JI)
   3483               Y(JI)=I
   3484             7 IJ=IJ+1
   3485               IF(IFLAG.EQ.2) GO TO 15
   3486               L1=NTTL2
   3487               DO 14 J=1,NTTL2
   3488               L1=L1+1
   3489               R1=X(L1)
   3490               R2=Y(L1)
   3491               X(L1)=X(J)
   3492               Y(L1)=Y(J)
   3493               X(J)=R1
   3494               Y(J)=R2
   3495            14 CONTINUE
   3496               R2= -X(N195)*FLOAT(NTTL2)
   3497               X(N194)=R2
   3498               IF(IFLAG.EQ.3) GO TO 15
   3499               NP=IFIX(SIMBW/X(N195))/2
   3500                R1=FSHIFT-X(N195)*NP
   3501                NSHIFT=IFIX(FSHIFT/X(N195))-NP+NTTL2
   3502                NP=NP*2
   3503                B1=R1-X(N195)/2.0
   3504                T1=R1+X(N195)/2.0
   3505           46   R3=R2+X(N195)*NSHIFT
   3506                WRITE(6,300) NSHIFT,NP,R1,R2,R3,B1,T1
   3507                IF(R3.GE.B1.AND.R3.LE.T1) GCTC 47
   3508                IF(R3.LT.B1) NSHIFT=NSHIFT+1
```

```
      IF(K2.GT.K1) NSHIFT=NSHIFT-1
      GOTO 48
47    IF(NSHIFT.GE.0) GOTO 49
      WRITE(6,53)
53    FORMAT(' THE VALUE OF NSHIFT IS NEGATIVE...NSHIFT SET TO 0')
      GOTO 52
49   L1=NSHIFT+NP
     IF(L1.LE.NTHROW) GO TO 50
     WRITE(6,51) NSHIFT,NP
  51 FORMAT(1H ,' THE SUM OF NSHIFT',I10,' AND NP',I10,' IS GT NTHROW....
    *NSHIFT SET TO 0')
52   NSHIFT=0
  50 CONTINUE
      X(N195)=DBLE(NP)
       X(N194)=-X(N195)*(NP/2)
      WRITE(6,600) NSHIFT,NP,K1,K2,X(N194)
      DO 425 J=1,NP
      NSHIFT=NSHIFT+1
      X(J)=X(NSHIFT)
      Y(J)=-Y(NSHIFT)
 425 CONTINUE
  15 CONTINUE
      Y(N193)=X(N193)
      Y(N194)=X(N194)
      Y(N195)=X(N195)
      RETURN
      END
```

SECTION 6

PERIPHERAL MODULES

This section includes all those modules that do not participate in the actual simulation process, but serve as the interface between the simulation and the user. In some cases this is as simple as outputting data to a disc file. In other cases the data is processed to extract some parameter, e.g. energy in a waveform, or to map data into a different representation space, e.g. the generation of a probability histogram. The following modules are included in this group:

CLINT
CUMDIS
ERGYCP
PLOTTR
PLTFMT
PTLIST
RTOPDB
SCANNR

Also included in this group is the module SPCAVG, which is located in Volume III.

The following peripheral modules are related to the bistatic target model and target imaging and are included in Volume I, Part 3:

ERRPRC
(BISTATIC ANTENNA INITIALIZATION)

SUBROUTINE CLINT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CLINT	PERIP.	302

2. PURPOSE:

This subroutine initializes the clutter model scatterers and parameters.

3. INPUT PARAMETERS:

Name	O/R	T	Description
LAMBDA	O	F	Wavelength of the transmitted waveform. This parameter is used to compute the doppler shift of weather or chaff clutter.
TCELL	R	F	Range dimension of clutter element.
IDMP	O	I	Output data print flag. IDMP ≠ 0 (a) Scatterer arrays are printed (CLINT) (b) Antenna Azimuth and Elevation arrays are printed (cluttr) IDMP = 1 Clutter impluse response is printed by (cluttr).
NRCS	R	I	Scatterer RCS probability distribution flag.
RWPH	O	F	Scatterer random walk maximum phase shift. Represents the random motion of cluttr from one transmit cycle to the next.
WNDVEL	O	F	Speed of the clutter volume due to wind, meters/second.

Name	O/R	T	Description
TI	R	F	Time increment between samples (used to compute number of storage cells required by clutter impulse response).
VELANG	O	F	Relative angle of clutter velocity vector with respect to the radar site.
RNEXT	O	F	Range dimension of the clutter volume in monoseconds.
RNØØØ	O	F	Clutter volume starting range.
AZEXT	O	F	Azimuth dimension of the clutter volume in degrees
AZØØØ	O	F	Clutter volume starting azimuth angle.
MM	O	I	Number of clutter volume azimuth increments.
ELEXT	O	F	Elevation dimension of the clutter volume in degrees.
ELØØØ	O	F	Clutter volume starting elevation angle.
NN	O	I	Number of clutter volume elevation increments.

4. CALLING SEQUENCES:

```
CALL  CLINT ($NNNN)
```

WHERE: NNNN Statement number for abnormal return

5. RESTRICTION, REQUIREMENTS, MISCELLANEOUS DATA:

a. The clutter model was split into two subroutines in order that the initialization process would be separate and easily changed to allow modeling of more complex clutter environments.

b. The total number of scatterers (NSCAT) is limited only by the amount of available disc space and is approximately equal to 160 times the number of blocks of available disc space. The number of scatterers is equal to NN times MM times KK, where KK = RNEXT/TCELL. Other than their product; NN, MM and KK have no limitations.

c. The random number generator function (RRAND) is called from this subroutine.

d. The clutter scatterer element arrays are stored in a random disc file, FC = 02, the record size is 500 words.

e. Flow Chart: Page 9-198

f. Cross Reference Table: Page 9-234

6. THEORY OF OPERATION

This program can be divided into two functional sections and each will be discussed in the following paragraph.

a. User parameter testing and clutter model parameter calculation. Refer to Figures CLINT - 1(a) and 1(b).

XVLANG The angle between the wind velocity vector and a radial line from the first clutter scatterer element (0,0,0) to the radar.

XVLANG = (AZØØØ - VELANG)/57.29578

DOPFRQ The maximum doppler frequency shift due to the wind.

DOPGRQ = -2.0 * WNDVEL/LAMBDA
If LAMBDA is zero this calculation is not performed and DOPFRQ is set to zero.

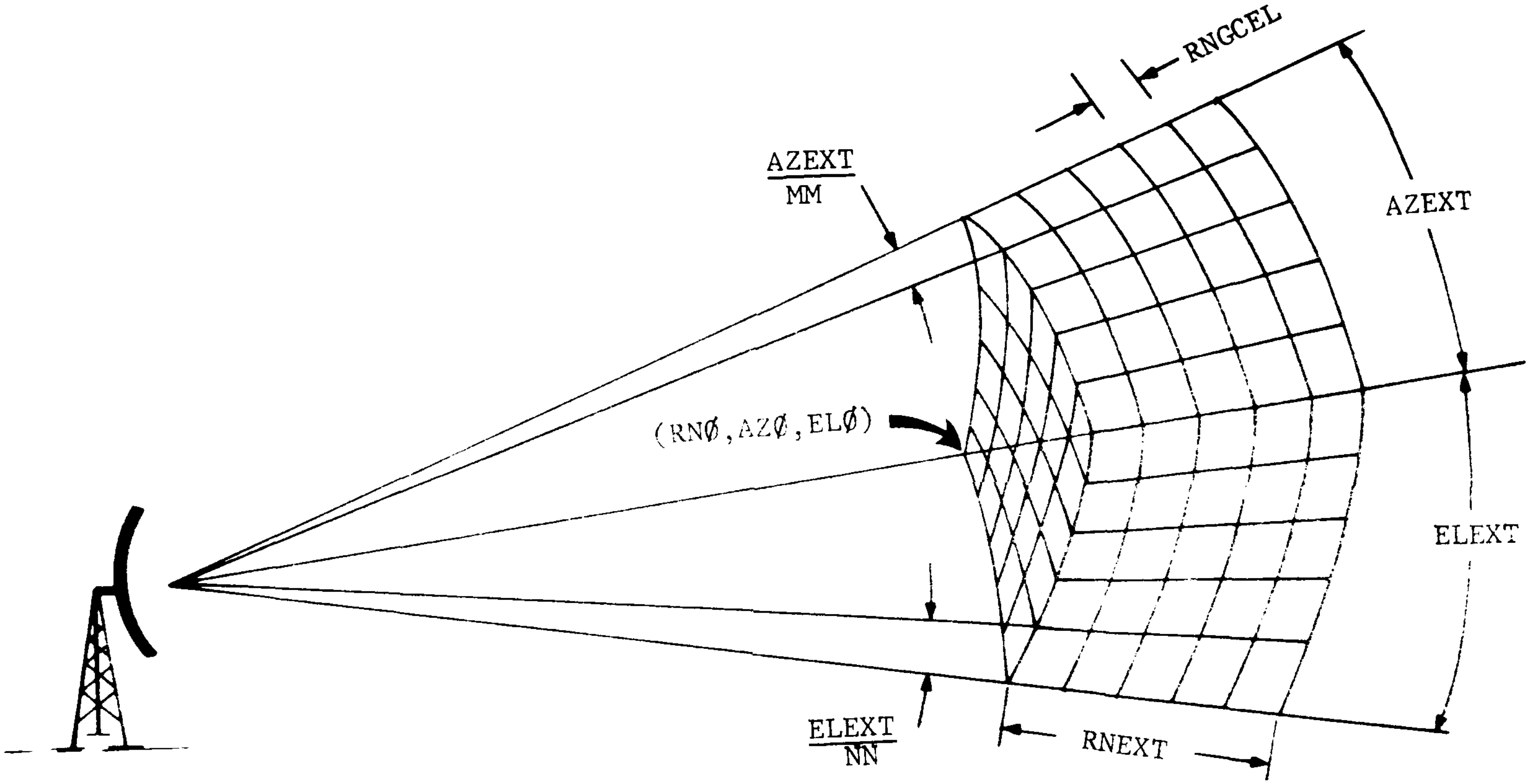


Figure CLINT-1a CLUTTER MODEL GEOMETRY

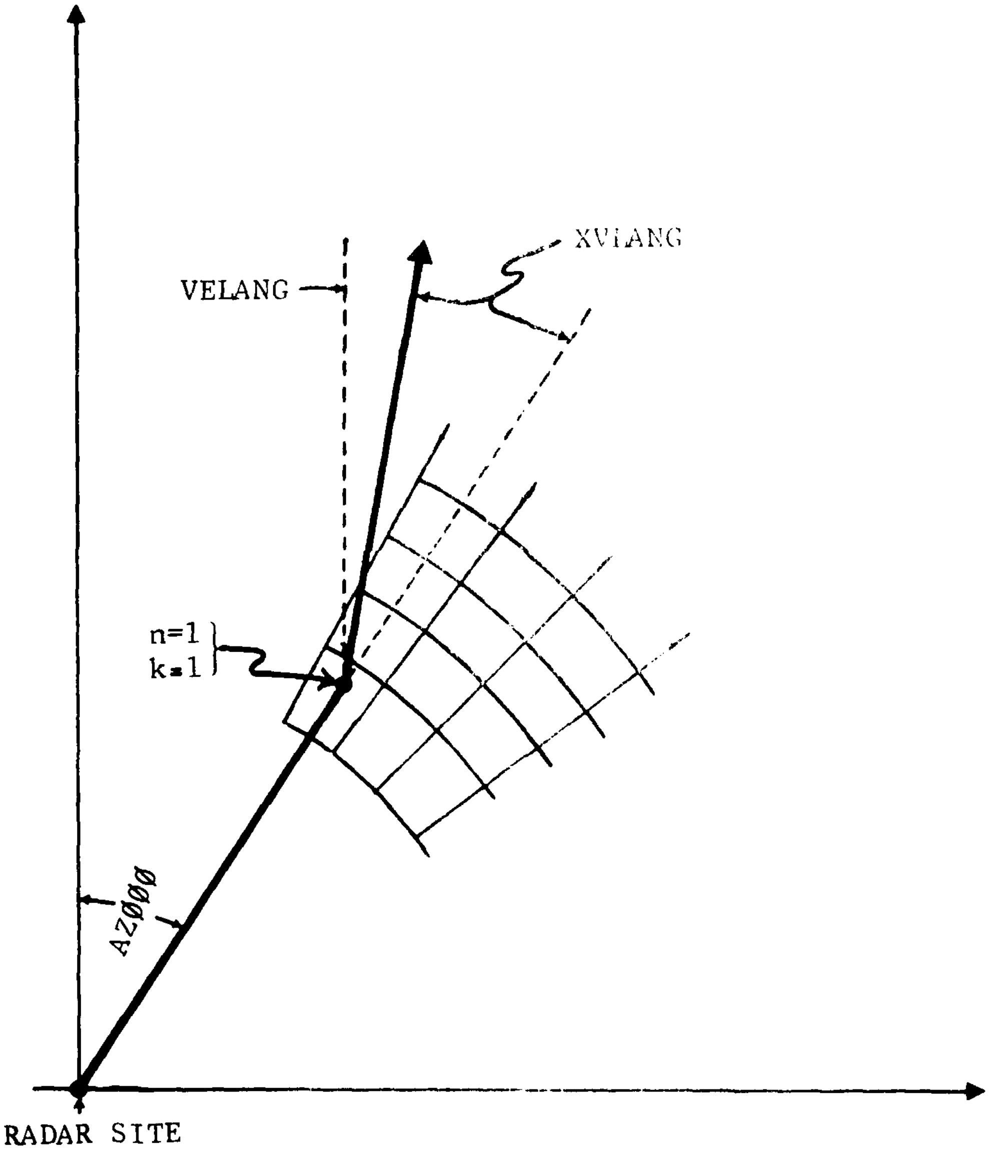


Figure CLINT-1b CLUTTER MODEL GEOMETRY (PLANE VIEW)

MM If this parameter was not specified it is set to 1.

NN If this parameter was not specified it is set to 1.

KCELL The total number of storage cells required to represent the clutter impulse response. If this parameter exceeds 4096 (half the available storage) then RNEXT is automatically reduced to make KCELL < 4096.

KCELL = RNEXT/TI

KK Number of clutter volume range increments.

KK = RNEXT/TCELL

NBLKS The number of random disc records required to store the clutter scatterer. This parameter is arbitrarily limited to 200 and can be increased providing the control cards for DSRN 02 are changed accordingly.

DELAZ Azimuth increment between clutter scatterers.

DELAZ = AZEXT/MM

DELEL Elevation increment between clutter scatterers.

DELEL = ELEXT/NN

ICFLG This parameter is set to 1 if the clutter model has been successfully initialized. This is tested before CLUTTR can be executed.

MODE This parameter is used to indicate if stationary (MODES) or time varying (MODE=2) clutter is to be modeled. When stationary clutter is modeled certain simplifications can be made in the processing performed in the cultter model routine CLUTTR.

b. Each clutter scatter element is represented by a radar cross section value and a phase angle. These two parameters are converted to rectanqular representation and stored in the arrays CLUX and CLUY.

CLUX(J) = A*RRAND(8)
CLUY(J) = A*DUM

where $A = \sqrt{RCS(J)} = \sqrt{RRAND(NRCS)}$
RRAND(8) = COS(θ)
DUM = SIN(θ)

θ is a sample of a random uniform distribution $300.0 < \theta \leq 0.0$

6-9

```
      SUBROUTINE CLINT(*)
      COMMON/BLK1/ CLUX(250),CLUY(250)
      COMMON/BLK2/ BK2(500)
      COMMON/BLKRND/ IDMR(8),DUM
      DIMENSION CSCAT(500)
      EQUIVALENCE (CLUX(1),CSCAT(1))
      EQUIVALENCE   (BK2( 13), LAMBDA       ),(BK2( 14), TCELL       ),
     *              (BK2( 21), IDMP         ),(BK2( 46), NRCS        ),
     *              (BK2( 47), NRWPH        ),(BK2( 48), RWPH        ),
     *              (BK2( 49), WNDVEL       ),(BK2( 50), VELANG      ),
     *              (BK2( 51), ENEXT        ),(BK2( 52), RNGDD       ),
```

C

```
     *              (BK2( 53), AZEXT       ),(BK2( 54), AZ000       ),
     *              (BK2( 55), MM          ),(BK2( 56), ELEXT       ),
     *              (BK2( 57), EL000       ),(BK2( 58), NN          ),
     *              (BK2(120), KK          ),(BK2(121), NBLKS       ),
     *              (BK2(122), MODE        ),(BK2(123), DELAZ       ),
     *              (BK2(124), DELEL       ),(BK2(125), XVLANG      ),
     *              (BK2(126), ICFLG       ),(BK2(127), DOPFRQ      )
     * ,            (BK2(128), KCELL       ),(BK2( 12), TI    )
      REAL LAMBDA
       CALL RANSIZ(02,500)
      XVLANG= (AZ000-VELANG)/57.29578
      IF(MM.GT.0) GO TO 100
      MM=1
      WRITE(6,95)
   95 FORMAT(1H0,' THE VARIABLE MM HAS BEEN SET TO 1 ' )
  100 IF(NN.GT.0) GO TO 200
      NN=1
      WRITE(6,195)
  195 FORMAT(1H0,' THE VARIABLE NN HAS BEEN SET TO 1 ' )
  200 CONTINUE
       KCELL=IFIX(RNEXT/TI)
       IF(KCELL.LT.8192) GOTO 50
       KCELL=8192
       RNEXT=TI*8192.0
 50    KK=IFIX(RNEXT/TCELL)
      IF(KK.GT.0) GO TO 211
      KK=1
  211 WRITE(6,212) KK
  212 FORMAT(1H0,' THE VARIABLE KK HAS BEEN SET TO',I5)
  CARD NO        ****                              CONTENTS                                ****

   5162                 NSCAT=MM*NN*KK
   5163                 NBLKS= NSCAT/250                                                   C
   5164                 IF(NBLKS*250.LT.NSCAT) NBLKS=NBLKS+1                               C
   5165                 IF(NBLKS.GT.200) GO TO 888
   5166                 DELAZ= AZEXT/FLOAT(MM)
   5167                 DELEL= ELEXT/FLOAT(NN)
   5168                 MODE=2
   5169                 IF(LAMBDA.NE.0.0) DOPFRQ=-2.0*WNDVEL/LAMBDA
   5170                 IF(DOPFRQ.EQ.0.0.AND.RWPH.EQ.0.0) MODE=1
   5171                 IPNT=1
   5172                 IREC=1
   5173                 DO 1000 K=1,KK
   5174                 DO 1010 N=1,NN
   5175                 DO 1020 M=1,MM
   5176                 IF(IPNT.LE.250) GO TO 950                                          C
   5177                 IF(IDMP.EQ.0) GO TO 940
   5178                 WRITE(6,930) ((CLDX(J),CLDY(J), J=1,IPNT)
   5179             930 FORMAT(1H ,6E20.6)
   5180             940 WRITE(2*IREC) CSCAT
   5181                  IREC=IREC+1
   5182                 IPNT=1
   5183            950   A=RRAND(NFLS)
   5184                 CLDX(IPNT)= A*RRAND(0)
   5185                 CLDY(IPNT)= A*DUM
   5186            1020  IPNT=IPNT+1
   5187            1010 CONTINUE
   5188            1000 CONTINUE
   5189                 IF(IPNT.EQ.1) GO TO 887
   5190                 WRITE(2*IREC) CSCAT
```

```
      IF(IDMP.NE.0) WRITE(6,930) (CLUX(J),CLUY(J), J=1,IFNT)
  887 ICFLG=1
      RETURN
  888 WRITE(6,889) KK,NN,MM,NSCAT
  889 FORMAT( 1H ,' THE PRODUCT OF KK=',I4,' NN=',I4,' MM=',I4,' IS TOO
     *LARGE,',I10,' THIS JOB WILL TERMINATE' )
      RETURN 1
      END
```

SUBROUTINE CUMDIS

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
CUMDIS	Peripheral	208,209
CUM2	Peripheral	Not User Referenced
OUTCUM	Peripheral	210,211
PDF	Peripheral	212,213

2. PURPOSE:

This subroutine processes a sequence of input samples in order to generate the probability density histogram and cumulative probability distribution.

3. INPUT PARAMETERS (CUMDIS):

Name	O/R	T	Description
TLIM	R	F	The upper limit of the histogram independent variable
BLIM	R	F	The lower limit of the histogram independent variable
NIXF	R	I	The number of elements in the histogram

INPUT PARAMETERS (OUTCUM):

Name	O/R	T	Description
NCPACK	R	I	The number of histogram elements combined to form one element of the output cumulative distribution

INPUT PARAMETERS (PDF):

Name	O/R	T	Description
NDPACK	R	I	The number of histogram elements combined to form one element of the output probability density.

4. CALLING SEQUENCES:

Initial histogram computation

CALL CUMDIS (DATAIN, XF)

Where: DATAIN contains the Input Waveform

XF contains the Output Histogram

Add additional data to histogram

CALL CUM2 (DATAIN, XF)

Where: DATAIN contains the Input Waveform

XF contains the Output Histogram

Cumulative distribution

CALL OUTCUM (DATOUT, XF)

Where: XF contains the Input Histogram

DATOUT contains the cumulative distribution

Probability density

CALL PDF (DATOUT, XF)

Where: XF contains the Input Histogram

DATOUT contains the probability density

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The number of points NIXF in the histogram XF cannot exceed 8000.

b. Any value of the input array which lies below the bound of BLIM will be placed in the lowest element of the histogram. Any value of the input array which lies above the bound of TLIM will be placed in the highest element of the histogram.

c. The variables NCPACK and NDPACK must be greater than or equal to 1.

d. Flow Chart: Page 9-152

e. Cross Reference Table: Page 9-228

6. THEORY OF OPERATION

a. The width of the histogram elements is determined using the following expression

DINC = (TLIM - BLIM) / NIXF

b. The value of each sample in the input array DATAIN is divided by DINC to determine the corresponding histogram element. The number of samples which lie within the bounds of that histogram element is incremented by one.

c. After the histogram is generated an entry through OUTCUM causes the cumulative distribution to be calculated and placed in array DATOUT. The number of histogram elements combined to generate one output sample is determined by the variable NCPACK.

d. If the density function is desired, entry is made through the entry point PDF. The number of histogram elements combined to generate one output sample is determined by the variable NDPACK.

6-14

```
      SUBROUTINE CUMDIS(DATAIN,XF)                                             C
C                                                                              C
      COMMON/BLK1/BK1(500)                                                     C
C                                                                              C
      DIMENSION DATAIN(1),DATOUT(1),XF(1)                                      C
     EQUIVALENCE  (BK1( 39), TLIM        ),
     1             (BK1( 40), BLIM        ), (BK1( 41), NIXF        ),         C
     2             (BK1( 42), NCPACK      ), (BK1( 43), NDPACK      )          C
      DATA N193,N194,N195,N196/-3,-2,-1,0/                                     C
C                                                                              C
C       CALCULATE HISTOGRAM ELEMENT WIDTH                                      C
      DINC = (TLIM-BLIM)/FLOAT(NIXF)                                           C
      LADD = 1- IFIX(BLIM/DINC)                                                C
      XF(N193)=BOOL(NIXF)                                                      C
      XF(N194)=BLIM                                                            C
      XF(N195)=DINC                                                            C
      XF(N196)=BOOL(LADD)                                                      C
C                                                                              C
      DO 20 J=1,NIXF                                                           C
   20 XF(J)=0.0                                                                C
      GO TO 1000                                                               C
      ENTRY CUM2(DATAIN,XF)                                                    C
      NIXF=IBOOL(XF(N193))                                                     C
      DINC=XF(N195)                                                            C
      LADD=IBOOL(XF(N196))                                                     C
C                                                                              C
C       ADD POINTS TO FREQ. DENSITY HISTOGRAM                                  C
```

```
 CARD NO        ****                            CONTENTS                               ****

   3944        C                                                                          C
   3945         1000 CONTINUE                                                             C
   3946              NDATA = ICELL(DATAIN(N193))                                          C
   3947              DINC=1.0/DINC                                                        C
   3948              DO 50 J =1,NDATA                                                     C
   3949              IADD=IFIX(DATAIN(J)*DINC) + LADD                                     C
   3950              IF(IADD.GT.NIXF) IADD=NIXF                                           C
   3951              IF(IADD.LT.1) IADD=1                                                 C
   3952           50 XF(IADD)=XF(IADD)+1.0                                                C
   3953        C                                                                          C
   3954              RETURN                                                               C
   3955        C                                                                          C
   3956              ENTRY OUTCUM(DATOUT,XF)                                              C
   3957        C                                                                          C
   3958        C        CALCULATE CUMULATIVE PROB. DISTRIBUTION                           C
   3959        C                                                                          C
   3960              IF(NCPACK.LE.0) NCPACK=1                                             C
   3961              NIXF=ICELL(XF(N193))                                                 C
   3962              DINC=XF(N195)                                                        C
   3963              KEND = NIXF/NCPACK                                                   C
   3964              DATOUT(N195) = DINC*FLOAT(NCPACK)                                    C
   3965              JJ = -NCPACK                                                         C
   3966              CUM=0.0                                                              C
   3967        C                                                                          C
   3968              DO 80 J=1,KEND                                                       C
   3969              JJ = JJ + NCPACK                                                     C
   3970              DO 70 K=1,NCPACK                                                     C
   3971           70 CUM=CUM+XF(K+JJ)                                                     C
   3972           80 DATOUT(J)=CUM                                                        C
```

6-15

```
  3973        C                                                                   C
  3974             ICON=0
  3975            GO TO 200                                                       C
  3976        C                                                                   C
  3977            ENTRY FDF(DATOUT,XF)                                            C
  3978        C                                                                   C
  3979             ICON=1
  3980             IF(NDPACK.LE.0) NDPACK=1
  3981             NIXF=BOOL(XF(N193))
  3982            DINC=XF(N195)                                                   C
  3983            KEND = NIXF/NDPACK                                              C
  3984            DATOUT(N195)=DINC*FLOAT(NDPACK)                                 C
  3985            JJ= -NDPACK                                                     C
  3986        C                                                                   C
  3987            CUM=0.0                                                         C
  3988            DO 100 J=1,KEND                                                 C
  3989            DEN=0.0                                                         C
  3990            JJ=JJ + NDPACK                                                  C
  3991            DO 90 K=1,NDPACK                                                C
  3992            DEN=DEN+XF(K+JJ)                                                C
  3993         90 CONTINUE                                                        C
  3994            CUM=CUM+DEN                                                     C
  3995            DATOUT(J)=DEN                                                   C
  3996        100 CONTINUE                                                        C
  3997        C                                                                   C
  3998        200 CONTINUE                                                        C
  3999             CUM=1.0/CUM
  4000             IF(ICON.EQ.1) CUM=CUM/DATOUT(N195)
  4001            DO 300 K=1,KEND                                                 C
02/11/75      INPUT LISTING                  AUTOFLOW CHART SET - FWD/SCL  RADSIM

 CARD NO      ****                        CONTENTS                               ****

  4002             DATOUT(K)=DATOUT(K)*CUM                                        C
  4003         300 CONTINUE                                                       C
  4004             DATOUT(N193) = BOOL(KEND)                                      C
  4005             DATOUT(N194)=XF(N194)                                          C
  4006             RETURN                                                         C
  4007             END                                                            C
```

SUBROUTINE ERGYCP

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
ERGYCP	Peripheral	118
ERGYRE	Peripheral	116,117

2. PURPOSE:

This subroutine computes the energy contained in a waveform.

3. INPUT PARAMETERS:

None

4. CALLING SEQUENCES:

Complex waveform

CALL ERGYCP (X,Y)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

$$\text{Energy} = \Delta \sum_{J=1}^{NPTS} X(J) **(J) + Y(J) * Y(J)$$

NPTS = Number of waveform samples

Δ = Independent variable increment between samples

Real Waveform

CALL ERGYRE (X)

Where: X contains the Input Waveform

$$\text{Energy} = \Delta \sum_{J=1}^{NPTS} X(J) * X(J)$$

a. The computed energy is printed on the output listing. The measurement unit is watt-nonoseconds, normally.

b. Flow Chart: Page 9-70

c. Cross Reference Table: Page 9-217.

```
      SUBROUTINE ENGYCF(X,Y)
      DIMENSION X(1),Y(1)
      DATA N193,N194,N195/-3,-2,-1/
      MODE=1
      GO TO 100
      ENTRY ENGYRF(X)
      MODE=0
100   NPTS=CODE(X(N193))
      DELT=X(N195)
      EGY=0.0
      DO 200 J=1,NPTS
      EGY=EGY+X(J)*X(J)
200   CONTINUE
      IF(MODE.EQ.0) GOTO 500
      DO 300 J=1,NPTS
      EGY=EGY+Y(J)*Y(J)
300   CONTINUE
500   EGY=EGY*DELT
      WRITE(6,1000) EGY
1000  FORMAT(' ENERGY IN THE WAVEFORM =',E15.5,' WATT-NANOSECONDS ' )
      RETURN
      END
```

SUBROUTINE PLOTTR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PLOTTR	PERIP	307,308,309,310

2. PURPOSE:

This subroutine converts the data contained in the input array into a form suitable for plotting by a HP9820 desk calculator. In addition, the output can also be listed on a TTY via TSS or punched on cards.

3. INPUT PARAMETERS:

Name	O/R	T	Description
SIVST	R	F	Value of the independent variable at the start of the plot
XIVRNG	R	F	Range of the independent variable which is to be plotted
LOGFLG	R	I	Control index which indicates the nature of the input data
NSKP	O	I	This variable controls the number of input samples shipped between each plotted point. For example, if NSKP=0 or 1, each point is plotted; if NSKP=2, every other point is plotted.
NAUTO	O	I	Control flag for automatic data scanner which determines the maximum and minimum values of the dependent variable. If NAUTO=0, the scanner determines the value of TH and TL. If NAUTO=1, the use provided input parameters TH and TL are used.
TH	O	F	Maximum dependent variable value to be plotted

Name	O/R	T	Description
IC	R	I	Logical unit designator for output file
VIL	0	C	Independent variable label
VDL	0	C	Dependent variable label
GLBL	0	C	Plot title and miscellaneous data

4. CALLING SEQUENCES:

CALL PLOTTR (DV)

Where: DV contains the Input Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. If an output file code (IC) is not specified then no data will be transferred and control will immediately be transferred back to the calling program.

b. If the user specified parameters, XIVST and SIVRNG, specify a plot that is outside the range of the input data array then the complete input array is processed and transferred to the output file.

c. If the file code is negative, specifying HP paper tape format, then all plot labels and miscellaneous data are suppressed. Only TH, TL. XIVOR, DELTA, and NPTS and the plot data are transferred.

d. All floating point output values are in E13.5 format.

e. Flow Chart: Page 9-90

f. Cross Reference Table: Page 9-219.

6. THEORY OF OPERATION

a. Upon entering the subroutine the parameter IC is tested to determine that IC is not zero and what operating mode is to be used. If IC 0 then the HP paper tape mode (ICON = 2) is selected. If IC 0 then the card image mode (ICON=1) is selected.

b. If (ICON=1) the following lines (records) are transmitted to the output file.

1. Header line "** Punch Card Data Output"
2. XIVST, XIVRNG, XIVFR (not used), LOGFLG, VIL (first 12 characters)
3. VIL (last 60 characters)
4. VDL (first 60 chara ters)
5. VDL (last 12 characters), GLBL (first 48 characters)

c. The input parameters, XVST and XIVRNG, are tested to verify that the range of the desired plot is within the limit of the input array, DV. If the desired plot extends past either end of DV, then the values of XIVST and XIVRNG are changed to be compatible with the available data. If the desired plot is totally outside the available data, then the complete input array is processed.

d. If NAUTO = 0 the input data is scanned to determine the maximum, TH, and the minimum, TL, values of the dependent variable to be plotted.

e. IF(ICON=1) the following lines (records) are transmitted to the output file.

1. GLBL (last 10 characters)
2. NSKP, NAUTO, IC

f. The following lines (records) are transmitted to the output file.

1. TH, TL, XIVOR, DELTA, NPTS
2. 5 dependent variable values
3. 5 dependent variable values

.

.

.

N. Last line of plot data

```
C PCDITR
      SUBROUTINE PCDITR(DV)
      COMMON/BEKI/ BKI(200),VIL(50),VDL(50),GLBL(50)
      EQUIVALENCE    ( BKI( 59), XIVST      ),( BKI( 60), XIVRNG     ),
     *               ( BKI( 61), XIVER      ),( BKI( 62), LOGFLG     ),
     *               ( BKI( 63), NSKP       ),( BKI( 64), NAUTO      ),
     *               ( BKI( 65), IH         ),( BKI( 66), IL         ),
     *               ( BKI(113), IC         )
      DATA N193,N194,N195,N196/-3,-2,-1,0/
      DIMENSION DV(1)
       IF(ABS(IC).GT.4) GOTO 600
       IF(IC) 500,600,10
   10  ICON=1
       GOTO 9
  500  IC=-IC
       ICON=2
       GOTO 12
    6      WRITE(IC,5)
     5 FORMAT(' **    PUNCH CARD DATA OUTPUT ')
C
      WRITE(IC,7) XIVST, XIVRNG, XIVER, LOGFLG, ( VIL(J), J=1,2 )
     7 FORMAT(1H ,3E15.5,I15,A6)
C
      WRITE(IC,6) ( VIL(J), J=3,12 )
     6 FORMAT(1H     ,10A6)
```

```
C
      WRITE(10,6) ( VDL(J), J=1,10 )
C
      WRITE(10,6) ( VDL(J), J=11,12 ), ( GLBL(J), J=1,8 )
C
   12  CONTINUE
      NTTL = IBOOL(DV(N193))
      XIVOR = DV(N194)
      DELTA = DV(N195)
      ENDDAT = XIVOR + DELTA*(NTTL-1)
       NST=IFIX((XIVST-XIVOR)/DELTA)+1
       IF(NST.LT.1) NST=1
       NSTOP=IFIX((XIVST-XIVOR+XIVRNG)/DELTA)+1
       IF(NSTOP.GT.NTTL) NSTOP=NTTL
       IF(NSTOP.GT.NST) GOTO 11
       NSTOP=NTTL
       NST=1
   11  CONTINUE
C
       XIVOR=XIVOR+(NST-1)*DELTA
       NPTS=(NSTOP-NST)/NSKP+1
       DELTA=DELTA*FLOAT(NSKP)
      IF(NAUTO.EQ.1) GO TO 100
       TH=DV(NST)
       TL=TH
       DO 50 J=NST,NSTOP
      TH=AMAX1(DV(J),TH)
      TL=AMIN1(DV(J),TL)
   50  CONTINUE
```

```
 CARD NO      ****                      CONTENTS

  2494        [illegible]  CONTINUE
  2495               IF(IECHO.EQ.2) GOTO 66'
  2496               WRITE(IO,60) (CERT(J),J=13,18),IH
  2497        60     FORMAT(2H *,6A6,E15.6)
  2498               WRITE(IO,10) NSKF,NAUTO,IC
  2499        10     FORMAT( [illegible] ,3I15)
  2500        [illegible]  WRITE(IO,22) IH,IL,XIVCF,DELTA,NPTS
  2501          22 FORMAT(1P4E15.5,I15)
  2502               NJMF=5*NSKF
  2503               DO 50 J=NST,NSTEP,NJMF
  2504              K = J+4*NSKF
  2505              WRITE(IO,25) (EV(N),N=J,K,NSKF)
  2506          25 FORMAT(1P5E15.5)
  2507          50 CONTINUE
  2508        C
  2509              RETURN
  2510        [illegible]  WRITE(6,601)
  2511        601    FORMAT(' FLUX DATA FILE IMPROPERLY DESIGNATED...NO DATA',
  2512             * ' TRANSFERED')
  2513               RETURN
  2514              END
```

SUBROUTINE PLTFMT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PLTFMT	PERIP	113

2. PURPOSE:

This subroutine is used to transform input data which is provided at equal increments of sine θ into an output array which is equally incremented in θ. The new points are determined using a parabolic interpolation. In addition, the input array is tested to determine if all input points are in visible space (independent variable between ± 1), and superflous data is eliminated.

3. INPUT PARAMETERS:

Name	O/R	T	Description
DELTHE	R	F	Output array independent variable increment
STARPT	0	F	Lower limit of the output array
STOPT	0	F	Upper limit of the output array

4. CALLING SEQUENCES:

CALL PLTFMT (GXIN, GYIN, GOUT, $nnnn)

Where: GXIN contains the Input E-field - Real Component

GYIN contains the Input E-field - Imaginary Component

GOUT contains the Output gain

nnnn is the statement number to which control is transferred when a non-recoverable error is detected.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The lower and upper limits of the output array must be specified in ascending order in order to be effective. If the upper limit is less than or equal to the lower limit or no limits are specified, all valid input data will be interpolated.

b. A double parabolic interpolation is performed to determine each output sample except in the vicinity of the end points.

c. Flow Chart: Page 9-175

d. Cross Reference Table: Page 9-231.

6. THEORY OF OPERATION

PLTFMT is intended primarily for converting antenna pattern data from equal increments of sine θ, generated by the Fast Fourier Transform, to equal increments of θ. This conversion is important in that it allows the pattern array to be stored as an array of gain points, equally spaced in angle. This eliminates the necessity of storing a gain-angle data pair, since the location of any point in angle can be determined from the ordinal number of the point, the origin of the array, and the angle increment. It is especially important in converting antenna pattern data for plots, since it is desirable that the points be equispaced in the independent variable direction. An additional feature of PLTFMT is that it will automatically eliminate the extraneous output from the Fast Fourier Transform, which is contained in the region commonly known as "invisible space", i.e., that region where the dependent variable lies outside $-1 \leq \text{sine } \theta \leq +1$. Furthermore, by specifying angle limits for the region of interest, superflous data is eliminated, thereby reducing processing time and storage requirements.

a. In the first section of the program the input array input parameters are initialized. The end points specified for the input data are examined to determine if they are realistic and within the boundaries of the input array. If not, one of three things happens. (1) If the specified end points exceed the array limits or ± 90 degrees, they are reduced to a useable value. (2) If the limits are equal or reversed, i.e., start point equals stop point or start point is greater than stop point, the limits are set to ± 90 degrees or to be array limits, whichever is less. (3) If the specified end points are in the proper order, but do not encompass any of the array data, the problem is terminated. If the adjusted end points do not encompass all of the input array, the portion of the input array to simplify processing and the array definition parameters are adjusted accordingly.

b. The output array elements are computed by interpolating between the input array points. A double parabolic interpolation is used except in the end segments where a single parabolic interpolation is used.

The double parabolic interpolation is performed by generating two quadratic equations, using three adjacent points to generate each equation. Assuming the points are labeled X1, X2, X3, and X4, and the required location is between X2 and X3, then the equations are generated using X1, X2, X3, and X2, X3, X4. Once the quadratic coefficients have been determined, the value of each quadratic at the point of interest is determined, and the two values are averaged to find the value of the dependent variable.

Within the end segments, adjacent points are not available to obtain two quadratic equations. Therefore, a single quadratic is generated and the necessary points are interp lated from this single equation. In all cases, if the point of interest falls exactly on a point in the input array, the actual value of that point is used.

```
                  SUBROUTINE  PLTEMT (GXIN,CYIN,GOUT,*)                          1940
           C**** THIS SUBROUTINE EXTRACTS THE SPECIFIED DATA FROM THE GENERATED    UC7PLT02
           C            ANTENNA PATTERN, DETERMINES THE ANGLE OF EACH GAIN POINT,  UC7PLT03
           C            AND INTERPOLATES BETWEEN DATA POINTS TO PROVIDE AN OUTPUT  UC7PLT04
           C            ARRAY OF GAIN POINTS EQUALLY SPACED IN ANGLE. ************** UC7PLT05
                  COMMON/ZEEP1/BKT(500)                                            UC7PLT06
                  DIMENSION GXIN(1),CYIN(1),GOUT(1)                                  2000
                  EQUIVALENCE  (BKT( 21), DUMY       ), (BKT( 33), DELTPT     )    UC7PLT09
                 *             (BKT( 34), STARPT     ), (BKT( 35), STOPT      )    UC7PLT10
                  DATA N193,N194,N195,RAD/-3,-2,-1,57.29578/                         2040
                  IF (DELTPT.GT.0.0) GO TO 7                                         2050
                  WRITE(6,119)                                                       2060
              119 FORMAT(*0AN ANGLE INCREMENT WAS NOT PROPERLY SPECIFIED FOR THE OUT  2070
                 *PUT ARRAY.  EXECUTION WILL NOT BE ATTEMPTED.*)                     2080
                  RETURN 1                                                           2090
                7 NPTS=IFIX(GXIN(N193))                                              2100
```

```
      ORIGIN=GXIN(N194)
      DELIN=GXIN(N195)
      IF(STARPT.GE.-90.0) GO TO 302
      STARPT=-90.0
      WRITE(6,106)
  106 FORMAT('0START POINT WAS DEFINED BELOW HORIZON.  START POINT HAS B
     *EEN REDEFINED TO -90 DEG.')
  302 IF(STOPT.LE.90.0) GO TO 303
      STOPT=90.0
      WRITE(6,111)
  111 FORMAT('0STOP POINT WAS DEFINED BELOW HORIZON.  STOP POINT HAS BEE
     *N REDEFINED TO 90 DEG.')
  303 IF(STARPT.LT.STOPT) GO TO 305
      STARPT=-90.0
      STOPT=90.0
      WRITE(6,109)
  109 FORMAT('0STARTING POINT SPECIFIED AS GREATER THAN OR EQUAL TO END
     * POINT.  ALL AVAILABLE VISIBLE SPACE WILL BE INTERPOLATED.')
  305 SINSTR=SIN(STARPT/RAD)
      SINSTP=SIN(STOPT/RAD)
  320 DELRAD=DELTHE /RAD
    9 NSTRT=IFIX((SINSTR-ORIGIN)/DELIN+0.5)
      IF(NSTRT.GE.2) GO TO 10
      STARPT=STARPT+DELTHE
      SINSTR=SIN(STARPT/RAD)
      GO TO 9
   10 NSTOP=IFIX((SINSTP-ORIGIN)/DELIN+0.5)
      IF(NSTOP+1.LE.NPTS) GO TO 15
      STOPT=STOPT-DELTHE
```

```
 CARD NO      ****                        CONTENTS                          ****

  4582              SINSTP=SIN(STOPT/RAD)                                   2400
  4583              GO TO 10                                                2410
  4584           15 STPRAD=STOPT/RAD                                        2420
  4585              ANGLE=STARPT/RAD                                        2430
  4586              VIV=ORIGIN+DELIN*FLOAT(NSTRT-2)                         2400
  4587              K=1                                                     2441
  4588              WRITE(6,1011) STOPT,STARPT,ORIGIN,DELIN,NPTS,NSTRT,NSTOP 2411
  4589         1011 FORMAT(1H ,4E15.7,3I10)                                 2412
  4590              DO 300 J=NSTRT,NSTOP                                    2450
  4591              X1=ARSIN(VIV)                                           2460
  4592              X2=ARSIN(VIV+DELIN)                                     2470
  4593              VIV=VIV+DELIN                                           2480
  4594              X3=ARSIN(VIV+DELIN)                                     2490
  4595              BREAK=(X3+X2)/2.0                                       2500
  4596              IF(BREAK.GT.STPRAD) BREAK=STPRAD                        2510
  4597              Y1=GXIN(J-1)                                            2520
  4598              Y2=GXIN(J)                                              2530
  4599              Y3=GXIN(J+1)                                            2540
  4600              MODE=1                                                  2550
  4601              X12=X1-X2                                               2560
  4602          350 YX12=(Y1-Y2)/X12                                        2570
  4603              C1=(YX12-((Y1-Y3)/(X1-X3)))/(X2-X3)                     2580
  4604              C2=YX12-C1*(X1+X2)                                      2590
  4605              C3=Y1-(C1*X1+C2)*X1                                     2600
  4606              GO TO (401,402), MODE                                   2610
  4607          401 CX1=C1                                                  2620
  4608              CX2=C2                                                  2630
  4609              CX3=C3                                                  2640
  4610              MODE=2                                                  2650
```

```
      Y1=CYIN(J-1)                                       6-27-a 2660
      Y2=CYIN(J)                                                2670
      Y3=CYIN(J+1)                                              2680
      GO TO 350                                                 2690
  402 CY1=C1                                                    2700
      CY2=C2                                                    2710
      CY3=C3                                                    2720
  450 IF(ANGLE.GT.BREAK) GO TO 299                              2730
      GR=(CX1*ANGLE+CX2)*ANGLE+CX3                              2740
      GI=(CY1*ANGLE+CY2)*ANGLE+CY3                              2750
      GOUT(K)=10.0*ALOG10(GR*GR+GI*GI)                          2750
      ANGLE=ANGLE+DELRAD                                        2770
      K=K+1                                                     2780
      GO TO 450                                                 2790
  299 IF(ANGLE.GT.STPRAD) GO TO 301                             2800
  300 CONTINUE                                                  2810
  301 GOUT(N193)=GOUT(K-1)                                      2820
      GOUT(N194)=STARPT                                         2830
      GOUT(N195)=DELTHE                                         2840
      RETURN                                                    2850
      END                                                       2860
```

SUBROUTINE PTLIST

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PTLIST	PERIP	303,304,305,306

2. PURPOSE:

This subroutine converts the data contained in the input array into a compact form for transmission to a remote processing station which has a CRT plotter.

3. INPUT PARAMETERS:

Name	O/R	T	Description
ST	R	F	Value of the independent variable at the start of the plot
RNG	R	F	Range of the independent variable which is to be plotted
LF	R	I	Control index which indicates the nature of the input data
NSKP	O	I	This variable controls the number of input samples skipped between each plotted point. For example, if NSKP = 0 or 1, each point is plotted; if NSKP = 2, every other point is plotted.
NAUTO	O	I	Control flag for automatic data scanner which determines the maximum and minimum values of the dependent variable. If NAUTO = 0 the scanner determines the value of TH and TL. If NAUTO = 1, the user provided input parameters TH and TL are used
TH	O	F	Maximum dependent variable value to be plotted

AD-A102 278   GENERAL DYNAMICS FORT WORTH TEX CONVAIR AEROSPACE DIV   F/6 17/9
ENDO ATMOSPHERIC-EXO ATMOSPHERIC RADAR MODELING. VOLUME II. PAR--ETC(U)
JUN 76   R J HANCOCK, F H CLEVELAND   F30602-73-C-0380
UNCLASSIFIED   RADC-TR-76-186-VOL-2-PT-1   NL

4 of 4
AD
A102278

END
DATE
FILMED
9 81
DTIC

Name	O/R	T	Description
TL	O	F	Minimum dependent variable value to be plotted
IFCODE	R	I	Logical unit designator for output file

4. CALLING SEQUENCES:

CALL PTLIST (DV)

Where: DV contains the Input Waveform

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. A number of tests are performed on the input data to ensure that a plot can be successfully completed. If all the necessary conditions are not satisfied, a condition code index is set and the plotter is bypassed with control returned to the calling program. The following is a list of the condition code values and the meaning of each:

CONDITION CODE INDEX	Error Message
1	The starting point, ST, is outside the input data array. (Too large)
2	The starting point, ST, is outside the input data array. (Too small)
3	The input data variables, ST, RNG, are improperly defined and result in a condition which places the stopping point before the starting point.
4	The user supplied value of TH is less than TL when NAUTO = 1.

b. The subroutine PACK is used to convert the fixed point output data into a form suitable for transmission to remote terminal via TSS.

c. If an output file code (IFCODE) is not specified then no data will be transferred and control will immediately be transferred back to the calling program.

d. Flow Chart: Page 9-82

e. Cross Reference Table: Page 9-218

6. THEORY OF OPERATION

a. Upon entering the subroutine the input parameters are first tested to verify that the range of the desired plot is within the limits of the input data array, DV.

b. If NAUTO = 0, the input data is scanned to determine the maximum, TH, and minimum, TL, values to be plotted. From these two variables the upper and lower limits of the plot are determined.

c. For magnitude data, TH and TL are tested to determine if the input data is bipolar or unipolar.

d. From TH and TL the dependent variable range of the plot is computed and the LSB and BIAS to be used in the conversion from floating point to fixed point is computed.

e. The following plot parameters are processed as one record:

1. TH - Maximum value of input array
2. TL - Minimum value of input array
3. IVST - First value of input array which was processed for output
4. NSTOP - Last value of input array which was processed for output
5. OR - Independent variable value for first output point
6. DEL - Independent variable spacing
7. RNG - Independent variable range of the output data

f. The following plot parameters are processed for output as one record:

1. BIAS - Number subtracted from input data before conversion to fixed point
2. XLSB - LSB of output data
3. - Not used
4. ITH - Largest dependent variable output value
5. ITL - Smallest dependent variable output value
6. NOUT - Number of output samples

g. The dependent variable values are processed for output. The mechanization equation for this conversion is the following:

$$IDAT=(DV(J)-BIAS)/XLSB \qquad IVST \leq J \leq NSTOP$$

The integer IDAT is then processed by the routine PACK which converts it into 2 characters and packs it into the array line which represents our output record and has the following structure:

>	Z	34 dependent variable samples	>	Z

header character pair   trailer character pair

```
                SUBROUTINE FILIST( DV)
                COMMON/BLK1/ BK1(200),VIL(50),VDL(50),CEDL(50)
                EQUIVALENCE   (  BK1( 59), ST          ),(  BK1( 60), KNG        ),
               *              (  BK1( 62), CF          ),(  BK1( 63), NSKP       ),
               *              (  BK1( 64), NAUTC       ),(  BK1( 65), TH         ),
               *              (  BK1( 66), TL          ),(  BK1(113),IFCODE   )
                DIMENSION DV(1),LINE(16)
                DATA N193,N194,N195,N196/-3,-2,-1,0/,AMIN/1.0E-10/
                 DATA LSIGN/07617700000000/
                 IF(IFCODE.EQ.0) GOTO 600
                IF(NSKP.LT.1) NSKP=1
                NTTL = IFIX(DV(N193))
                CF = DV(N194)
                DEL = DV(N195)
                T1=ST*KNG
                T2=CF*DEL*FLOAT(NTTL-1)
                IF(ST.GE.CF) GO TO 50
                KNG=KNG-CF*ST
                ST=0
             50 CONTINUE
                IF(T2.GT.TH) GO TO 70
                TH=T2+1
             70 CONTINUE
                IVSL = IFIX(T(I-CF) [illegible]
                [illegible]
```

```
      IF(IVST.GE.NTTL) NCTRL =1
      NSTOP = IFIX((ST-OR+RNG)/DEL) + 1
       OR=OF+(IVST-1)*DEL
      IF(NSTOP.GT.NTTL) NSTOP = NTTL
      IF(NSTOP.LE.IVST) NCTRL = 4
      IF(NCTRL.NE. 0) GO TO 1100
      IF(LF.LT.2) GO TO 40
      DO 45 J=IVST,NSTOP
       X=AMAX1(ABS(DV(J)),XMINDB)
       IF(LF.EQ.2) GO TO 41
       DV(J)=10.0*ALOG10(X)
       GO TO 42
   41 DV(J)=ALOG10(X)
   42 CONTINUE
   45 CONTINUE
   40 CONTINUE
        TH = DV(IVST)
        TL = TH
        IF(NCTRL.NE.0) GO TO 1100
        DO 100 J=IVST,NSTOP
        TH = AMAX1(DV(J),TH)
        TL = AMIN1(DV(J),TL)
  100 CONTINUE
        IF(TH.GT.TL) GO TO 2001
        NCTRL=5
        GO TO 1100
 2001 CONTINUE
          DEL=DEL*FLOAT(NSKP)
          NOUT=(NSTOP-IVST)/NSKP+1
```

```
CARD NO    ****                           CONTENTS                                        ****

 2320            LINE(1)=BCOL(TH)
 2321            LINE(2)=BCOL(TL)
 2322            LINE(3)=IVST
 2323            LINE(4)=NSTOP
 2324            LINE(5)=BCOL(GR)
 2325            LINE(6)=BCOL(DEL)
 2326            LINE(7)=BCOL(RNG)
 2327            WRITE(IFCODE) LINE
 2328            IF(LP.EQ.1.OR.LP.EQ.3) GOTO 700
 2329           IF(TH.LE.0.0) ICON=1
 2330           IF(TL.GE.0.0) ICON=2
 2331           IF(TH.GT.0.0.AND.TL.LT.0.0) ICON= 3
 2332           FTH= AMAX1(ABS(TH),ABS(TL))
 2333           FTL= ALOG10(FTH)
 2334           ITEST = IFIX(FTL)
 2335           IF(FTL.LT.0.0) GO TO 150
 2336           X=FTH/(10.0**ITEST)
 2337           GO TO 160
 2338       150 X=FTH*(10.0**(IABS(ITEST)+1))
 2339       160 CONTINUE
 2340            N=20
 2341           IF(X.GE.1.0.AND.X.LE.1.8) N=2
 2342           IF(X.GT.1.8.AND.X.LE.3.5) N=4
 2343           IF(X.GT.3.5.AND.X.LE.4.5) N=5
 2344           IF(X.GT.4.5.AND.X.LE.7.0) N=8
 2345           IF(X.GT.7.0.AND.X.LE.9.0)N=10
 2346           IF(FTL.LT.0.0) GO TO 500
 2347           PIP=FLOAT(N)*(10.0**ITEST)
 2348           GO TO 500
 350 PTH = FLOAT(N) /(10.0**(IABS(ITEST)+1))
 300 CONTINUE
       ALSD=PTH/4000.0
       BIAS=0.0
     GO TO(399,401,402),ICON
 399 AMX = 0.0
     AMN =-TH
     AMD =-TH/2.0
     DINC= TH/100.0
     GO TO 404
 401 AMX = TH
     AMN = 0.0
     AMD = TH/2.0
     DINC= TH/100.0
     GO TO 404
 402 AMX = TH
     AMN =-TH
     AMD = 0
     DINC= TH/50.0
     GO TO 404
 700 AMD=ABS(TH-TL)
     N = (IFIX (AMD/10.0))*10
     IF(AMD.GT.100.0) N=100
     AMX = (FLOAT(IFIX(TH/10.0)))*10.0
     IF(AMX.LT.TH) AMX=AMX+10.0
     AMN= AMX-FLOAT(N)
     IF(AMN.GT.TL.AND.N.LT.100) N=N+10
     IF(N.EQ.90) N=100
     IF(N.EQ.30) N=40
```

```
CARD NO      ****                          CONTENTS

 2378              IF(N.EQ.60.OR.N.EQ.70) N=60
 2379              BIAS= AMX-FLOAT(N)
 2380               XLSB=0.05
 2381               IF(N.LE.40) XLSB=0.01
 2382               IF(N.GT.40.AND.N.LE.60) XLSB=0.02
 2383          404 IF(NCTRL.NE.0)GO TO 1100
 2384               ITL=(TL-BIAS)/XLSB
 2385               ITH=(TH-BIAS)/XLSB
 2386               LINE(1)=BOOL(BIAS)
 2387               LINE(2)=BOOL(XLSB)
 2388               LINE(3)=N
 2389               LINE(4)=ITH
 2390               LINE(5)=ITL
 2391               LINE(6)=NCUT
 2392               WRITE(IFCODE) LINE
 2393               LINE(1)=LSTCH
 2394               LINE(10)=LSTCH
 2395          502  IWD=1
 2396               IBIT=18
 2397               CALL PACK(NCUT,IWD,IBIT,LINE,$1100)
 2398              DO 400 J=IVST,NSTOP,NSKP
 2399               IDAT=(DV(J)-BIAS)/XLSB
 2400               IF(IDAT.GT.4095) IDAT=4095
 2401               IF(IDAT.LT.-4095) IDAT=-4095
 2402               CALL PACK(IDAT,IWD,IBIT,LINE,$500)
 2403               GO TO 400
 2404          500  WRITE(IFCODE) LINE
 2405               IWD=1
 2406               IBIT=18
```

```
      [illegible] CONTINUE
          IF(IW[illegible]) RETURN
          WRITE([illegible]) LINE
          RETURN
      [illegible] WRITE([illegible]) NCTRL
          RETURN
      [illegible] FORMAT(' [illegible] NOT ATTEMPTED DUE TO CONDITION CODE =',I4)
      [illegible] WRITE([illegible])
      [illegible] FORMAT(' [illegible] DATA [illegible] FILE IMPROPERLY [illegible]...',
         * ' [illegible]')
          END
```

SUBROUTINE RTOPDB

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
RTOPDB	Peripheral	103
RTOPM	Peripheral	110
RTOPM2	Peripheral	111
XYTOM	Peripheral	105
XYTOM2	Peripheral	106
XYTODB	Peripheral	104,108

2. PURPOSE:

This subroutine performs a rectangular to polar conversation on the input arrays. Depending on the entry point used, the output is in two arrays in the form of linear magnitude and phase (RTOPM), magnitude squared and phase (RTOPM2) or magnitude in decibles and phase (RTOPDB); or in one array of linear magnitude (XYTOM), linear magnitude squared (XYTOM2) or magnitude in decibels (XYTODB).

3. INPUT PARAMETERS: None

4. CALLING SEQUENCES:

Phase and decibel magnitude

CALL RTOPDB (X,Y,M,P)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform

P contains the Output Waveform - P

$$M(J) = 10.0 * \mathrm{ALOG10} \left[X(J) * X(J) + Y(J) * Y(J) \right]$$

$$P(J) = \frac{180.0}{\pi} * \mathrm{ATAN2}\ (Y(J),\ X(J))$$

Phase and linear magnitude

CALL RTCPM (X,Y,M,P)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform - M

P contains the Output Waveform - P

$$M(J) = \sqrt{X(J) * X(J) + Y(J) * Y(J)}$$

$$P(J) = \frac{180.0}{\pi} * \text{ATAN2}\ (Y(J),X(J))$$

Phase and magnitude squared

CALL RTOPM2 (X,Y,M,P)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform - M

P contains the Output Waveform - P

$$M(J) = X(J) * X(J) + Y(J) * Y(J)$$

$$P(J) = \frac{180.0}{\pi} * \text{ATAN2}\ (Y(J), X(J))$$

Linear magnitude and no phase

CALL XYTOM (X, Y, M)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform

$$M(J) = \sqrt{X(J) * X(J) + Y(J) * Y(J)}$$

Magnitude squared and no phase

CALL XYTOM2 (X,Y,M)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform

$$M(J) = X(J) * X(J) + Y(J) * Y(J)$$

Magnitude in dB and no phase

CALL XYTODB (X,Y,M)

Where: X contains the Input Waveform - R

Y contains the Input Waveform - I

M contains the Output Waveform

$$M(J) = 10.0 * \mathrm{ALOGIO} \left[X(J) * X(J) + Y(J) * Y(J) \right]$$

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. For entry points RTOPDB and XYTODB the minimum output value is -200 dB.

b. The range of the output phase angle is from -190° to $+170^{\circ}$. The break point where phase "rolls over" was chosen to be 170° instead of 180° because many waveforms have a large number of samples around 180° and each time the waveform's phase changes from 180 -E to 180 +E the output phase would change from 180-E to -180+E. This results in a phase plot with many undesirable and unnecessary discontinuities.

c. Flow Chart: Page 9-156

d. Cross Reference Table: Page 9-228.

```
      SUBROUTINE RTOPDB(X,Y,M,P)                                        R
       COMMON/BLK1/ BK1(500)
       EQUIVALENCE (BK1( 36), NAVG)
      DIMENSION X(1),Y(1),M(1),P(1)                                     R
      REAL M                                                            R
      DATA N193,N196,N4/-3,0,4/                                         R
      DATA AI/57.29578/                                                 R
      ICON = 2                                                          R
      NPHAS=1                                                           R
      GO TO 10                                                          R
      ENTRY RTOFM(X,Y,M,P)                                              R
      ICON = 2                                                          R
      NPHAS=1                                                           R
      GO TO 10                                                          R
      ENTRY RTOFM2(X,Y,M,P)                                             R
      ICON = 0                                                          R
      NPHAS=1                                                           R
      GO TO 10                                                          R
      ENTRY XYTOM(X,Y,M)                                                R
      ICON=1                                                            R
      NPHAS=0                                                           R
      GO TO 10                                                          R
      ENTRY XYTOM2(X,Y,M)                                               R
      ICON=0                                                           R
      NPHAS=0                                                          R
      GO TO 10                                                         R
      ENTRY XYTODB(X,Y,M)                                              R
      ICON=2                                                           R
      NPHAS=0                                                          R
C                                                                      R
   10 CONTINUE                                                         R
      N=IBCOL(X(N193))                                                 R
      CALL DBLKX(N193,N4,X,M)                                          R
      IF(NPHAS.EQ.1) CALL DBLKX(N193,N4,X,P)                           R
       IF(NPHAS.EQ.0.AND.NAVG.GT.1) GOTO 200
      IF(NPHAS.EQ.1) P(N196)=0.0                                       R
C***********************************************************************R
      DO 100 J=1,N                                                     R
      IF(NPHAS.EQ.0) GO TO 50                                          R
      IF(X(J).EQ.0.0.AND.Y(J).EQ.0.0) GO TO 40                         R
      P(J) = AI * ATAN2(Y(J),X(J))                                     R
       IF(P(J).GT.170.0) P(J)=P(J)-360.0
      GO TO 50                                                         R
   40 P(J) = P(J-1)                                                    R
   50 CONTINUE                                                         R
      M(J)= X(J)*X(J)+Y(J)*Y(J)                                        R
      IF(ICON.EQ.1) M(J)=SQRT(M(J))                                    R
      IF(ICON.NE.2) GO TO 100                                          R
      IF(M(J).LT.1.0E-20) M(J)=1.0E-20
      M(J)= 10.0*ALOG10(M(J))                                          R
  100 CONTINUE                                                         R
      RETURN                                                           R
```

```
CARD NO        ****                       CONTENTS                           ****

               200   DO 210 J=1,N
                     M(J)=X(J)*X(J)+Y(J)*Y(J)
               210   CONTINUE
                     AV=0.0
                     DO 220 J=1,NAVG
                     AV=AV+M(J)
               220   CONTINUE
                     N=N-NAVG
                     XAVG=1.0/FLOAT(NAVG)
                     DO 230 J=1,N
                     DUM=M(J)
                     M(J)=AV*XAVG
                     IF(ICON.EQ.1) M(J)=SQRT(M(J))
                     IF(ICON.NE.2) GOTO 225
                    IF(M(J).LT.1.0E-20) M(J)=1.0E-20
                    M(J)= 10.0*ALOG10(M(J))                                      R
               225   AV=AV-DUM+M(J+NAVG)
               230   CONTINUE
                     M(N193)=BOOL(N)
                     RETURN
                     END
```

SUBROUTINE SCANNR

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
SCANNR	PERIP	313

2. PURPOSE:

This subroutine scans an antenna pattern in the form of an input array and determines the location and gain of (1) the main lobe, (2) the highest ten sidelobes, and (3) the first twenty sidelobes on either side of the main lobe. In addition, the first minimum on either side of the main lobe is determined.

3. INPUT PARAMETERS:

None

4. CALLING SEQUENCES:

CALL SCANNR (DV)

Where: DV contains the Input Waveform in dB gain

5. RESTRICTION, REQUIREMENTS, MISCELLANEOUS DATA

a. All output data from this subroutine is in the form of a printed listing which is formatted for ease of interpretation.

b. If a vestigial lobe exists, the minimum between it and the main lobe will be detected as the main lobe null.

c. Flow Chart: Page 9-161

d. Cross Reference Table: Page 9-229

6. THEORY OF OPERATION

a. The antenna pattern data is scanned from left to right (increasing θ) and the gain maxima are stored in a 22 element array. Within this array, the maxima are ordered according to gain with the highest in element 1 and the lowest in element 22.

b. The pattern data to the right of the main lobe is scanned to locate the main lobe null and the first twenty sidelobes. After this data is processed and printed, a third scan is made to the left of the main lobe for the first null and the first twenty sidelobes.

c. The logical equation used to determine the peaks of the lobes is as follows:

Given three adjacent points DV(NTHPT-1), DV(NTHPT), and DV(NTHPT+1) the following difference parameters are computed.

$$\begin{aligned} \text{DELDV1} &= \text{DV(NTHPT)} - \text{DV(NTHPT-1)} \\ \text{DELDV2} &= \text{DV(NTHPT+1)} - \text{DV(NTHPT)} \end{aligned}$$

then if DELDV1 ≥ 0 and DELDV2 < 0, the point DV(NTHPT) is considered to be a maxima, i.e., the peak of a lobe.

d. In order to find the nulls of the main lobe, the difference parameters defined in 6(c) are tested for the following condition:

$$\text{DELDV1} < 0 \quad \text{and} \quad \text{DELDV2} \geq 0$$

If this condition is satisfied then DV(NTHPT) is considered to be a null. After the first null is found, a flag is set which causes the null test to be bypassed for the remainder of the scan on that side of the main lobe. It should be noted that the null logic will consider the minimum between the main lobe and a vestigial sidelobe (if one exists) to be the main lobe null.

```
      SUBROUTINE SCANNER(DV)                                      6.42 UC7SCN01
C                                                                      UC7SCN02
C**** THIS SUBROUTINE SCANS THE UNINTERPOLATED ANTENNA PATTERN DATA AND UC7SCN03
C  PRINTS OUT MAIN LOBE NULL POINTS AND SIGNIFICANT SIDE-LOBES. ******* UC7SCN04
C                                                                      UC7SCN05
      DIMENSION LOBEPT(22), ALOBGN(22), DV(1)                          UC7SCN06
      DATA N193,N194,N195,N196/-3,-2,-1,0/                             UC7SCN07
      EQUIVALENCE (NPTS,XPTS)                                          UC7SCN08
C                                                                      UC7SCN09
      WRITE(6,1)                                                       UC7SCN10
    1 FORMAT (1H0/47X,'* * * * ANTENNA LOBE SCANNER * * * *' / / / )   UC7SCN11
      NPTS = IFIX(DV(N193))                                            UC7SCN22
      XBEG = DV(N194)                                                  UC7SCN13
      PTSPC = DV(N195)                                                 UC7SCN14
C**** INITIALIZE LOBE STORAGE ARRAY *********************************** UC7SCN15
      DO 10 N=1,22                                                     UC7SCN16
      LOBEPT(N) = 0                                                    UC7SCN17
   10 ALOBGN(N) = -200.0                                               UC7SCN18
C                                                                      UC7SCN19
C**** FIND MAXIMA ****************************************************** UC7SCN37
      DO 70 NTHPT=1,NPTS
      DELDVI = DV(NTHPT) - DV(NTHPT-1)                                 UC7SCN39
```

```
            DELDV2 = DV(NTHPT+1) - DV(NTHPT)                                    UC7SCN40
            IF( DELDV1 .LT. 0.0 .OR. DELDV2 .LE. 0.0 ) GO TO 70                 UC7SCN41
      C                                                                         UC7SCN42
            LOBPT(22) = NTHPT                                                   UC7SCN43
            ALOBGN(22) = DV(NTHPT)                                              UC7SCN44
      C                                                                         UC7SCN45
      C**** ORDER TWENTY-ONE HIGHEST MAXIMA *************************************** UC7S
            DO 50 NF=1,21                                                       UC7SCN47
            IF( ALOBGN(23-NF) .LE. ALOBGN(22-NF) ) GO TO 70                     UC7SCN48
            ANTREG = ALOBGN(23-NF)                                              UC7SCN49
            ALOBGN(23-NF) = ALOBGN(22-NF)                                       UC7SCN50
            ALOBGN(22-NF) = ANTREG                                              UC7SCN52
            NLOBPT = LOBPT(23-NF)                                               UC7SCN52
            LOBPT(23-NF) = LOBPT(22-NF)                                         UC7SCN53
         50 LOBPT(22-NF) = NLOBPT                                               UC7SCN54
      C                                                                         UC7SCN55
         70 CONTINUE                                                            UC7SCN56
      C                                                                         UC7SCN57
      C**** COMPUTE MAIN LOBE ANGLE AND OUTPUT ****************************** UC7SCN58
            ANGMN = XERO + (LOBPT(1)-1)*FTSPC                                   UC7SCN59
            WRITE (6,150) ALOBGN(1)
        150 FORMAT(' THE MAIN LOBE GAIN IS ',F15.7)
      C                                                                         UC7SCN61
            WRITE(6,72) ANGMN                                                   UC7SCN62
         72 FORMAT( 1H0, 34X, 'LOCATION AND RELATIVE GAIN OF HIGHEST 20 SIDE LOUC7SCN63
           +BES' / 35X,'RANK',17X,'REL. GAIN',16X,'ANGLE' / 5X,'( DEG )',16X,UC7SCN64
           +'( DEG )' // 35X,'MAIN',20X,'0.0',17X,F15.7)                         UC7SCN6
      C                                                                         UC7SCN66
      C**** COMPUTE SIDE-LOBE RELATIVE GAIN AND ANGLE, AND OUTPUT *********** UC7SCN67
```

```
 CARD NO      ****                          CONTENTS                              ****

  4176              DO 75 N=2,21                                               UC7SCN68
  4177              GAIN = ALOBGN(N) - ALOBGN(1)                               UC7SCN69
  4178              NK = N-1                                                   UC7SCN70
  4179              ANGLE = XORG + (LOBPT(N)-1)*PTSPC                          UC7SCN71
  4180              WRITE(6,74) NK, GAIN, ANGLE                                UC7SCN73
  4181           74 FORMAT ( 36X, I2, 18X, F13.7, 14X,F13.7)                     UC7SCN
  4182           75 CONTINUE                                                   UC7SCN75
  4183        C                                                                UC7SCN76
  4184              WRITE(6,76)                                                UC7SCN77
  4185           76 FORMAT(1H1,21X,'MAIN LOBE NULL AND FIRST 20 SIDE-LOBES TO RIGHT OFUC7SCN78
  4186             * (ABOVE) MAIN LOBE'                                        UC7SCN79
  4187             */35X,'RANK',13X,'REL. GAIN',13X,'TRUE ANGLE',13X,'REL. ANGLE'
  4188             */ 37X,'( DB )',16X,'( DEG )',16X,'( DEG )' /)
  4189        C                                                                UC7SCN82
  4190        C**** INITIALIZE RIGHTHAND SCANNER ********************************** UC7SCN83
  4191              NULFLG = 0                                                 UC7SCN84
  4192              NLB = 1                                                    UC7SCN85
  4193              NWSTRT = LOBPT(1)+1                                        UC7SCN86
  4194        C**** FIND SIDE-LOBES TO RIGHT OF MAIN LOBE ************************** UC7SCN87
  4195              DO 85 NTHPT=NWSTRT,NPTS                                    UC7SCN88
  4196              DELDV1 = DV(NTHPT) - DV(NTHPT-1)                           UC7SCN89
  4197              DELDV2 = DV(NTHPT+1) - DV(NTHPT)                           UC7SCN90
  4198              IF (DELDV1 .GE. 0.0 .AND. DELDV2 .LT. 0.0) GO TO [illegible] UC7SCN91
  4199        C**** FIND RIGHTHAND MAIN LOBE NULL ********************************* UC7SCN92
  4200              IF (NULFLG.EQ.1 .OR. DELDV1.GE.0.0 .OR. DELDV2.LT.0.0) GO TO 85   UC7SCN93
  4201              NULFLG = 1                                                 UC7SCN94
  4202        C**** COMPUTE ANGLE OF RIGHTHAND MAINLOBE NULL ********************** UC7SCN95
  4203              AMNLRT = XORG + (NTHPT-1)*PTSPC                             UC7SCN9
  4204              WRITE (6, 84) AMNLRT                                       UC7SCN98
```

```
   84 FORMAT(38X,'MAIN LOBE',9X,'** NULL **',13X,F13.7)
      GO TO 85                                                          UC7SCN00
C                                                                       UC7SCN01
C**** COMPUTE RELATIVE GAIN AND RELATIVE ANGLE OF RIGHTHAND SIDE-LOBES  UC7SCN02
   82 CONTINUE                                                          UC7SCN03
      GAIN = DV(NTHPT) - ALOPGN(1)                                      UC7SCN04
      ANGLB = XORG + (NTHPT-1)*PTSPC                                    UC7SCN05
      RELANG=ANGLB - ANGMN
C                                                                       UC7SCN07
      WRITE(6,80) NLB, GAIN, ANGLB, RELANG
   80 FORMAT ( 39X,I3,3(13X,F13.7))
      NLB = NLB+1                                                       UC7SCN10
C**** EXIT AFTER TWENTY SIDE-LOBES OR DROP THROUGH AT END OF DATA ***** UC7SCN11
      IF( NLB .GT. 20 ) GO TO 87                                        UC7SCN12
C                                                                       UC7SCN13
   85 CONTINUE                                                          UC7SCN14
      WRITE(6,86)                                                       UC7SCN15
   86 FORMAT( 45X, 'NO FURTHER SIDE LOBES ON THIS SIDE' )               UC7SCN16
C                                                                       UC7SCN17
   87 WRITE(6,88)                                                       UC7SCN18
   88 FORMAT(1H0/22X,'MAIN LOBE NULL AND FIRST 20 SIDE-LOBES TO LEFT OF UC7SCN19
     *(BELOW) MAIN LOBE'                                                UC7SCN20
     */38X,'RANK',13X,'REL. GAIN',13X,'TRUE ANGLE',13X,'REL. ANGLE'
     */57X,'( DB )',16X,'( DEG )',16X,'( DEG )' / )
C                                                                       UC7SCN23
C**** INITIALIZE LEFTHAND SCANNER ************************************** UC7SCN24
      NWSTRT = LOBPT(1)                                                 UC7SCN25
      NWSTP = NWSTRT-2
      NLB = 1                                                           UC7SCN27
```

```
 CARD NO      ****                          CONTENTS                                   ****

  4234               NULFLG = 0                                                       UC7SCN28
  4235          C                                                                     UC7SCN29
  4236          C**** FIND SIDE-LOBES TO LEFT OF MAIN LOBE *************************** UC7SCN30
  4237               DO 93 NTHPT=1,NWSTP                                              UC7SCN31
  4238               NXPT = NWSTRT - NTHPT                                            UC7SCN32
  4239               DELDV1 = DV(NXPT) - DV(NXPT-1)                                   UC7SCN33
  4240               DELDV2 = DV(NXPT+1) - DV(NXPT)                                   UC7SCN34
  4241               IF (DELDV1 .LE. 0.0 .AND. DELDV2 .LT. 0.0) GO TO 90              UC7SCN35
  4242          C**** FIND LEFTHAND MAIN LOBE NULL *********************************** UC7SCN36
  4243               IF (NULFLG.EQ.1 .OR. DELDV1.GE.0.0 .OR. DELDV2.LT.0.0) GO TO 93  UC7SCN37
  4244          C**** COMPUTE ANGLE OF LEFTHAND MAINLOBE NULL ************************ UC7SCN38
  4245               AMNLFT = XORG + (NXPT-1)*PTSPC                                    UC7SCN3
  4246               NULFLG = 1                                                       UC7SCN41
  4247               WRITE (6, 84) AMNLFT                                             UC7SCN42
  4248               GO TO 93                                                         UC7SCN43
  4249          C                                                                     UC7SCN44
  4250          C**** COMPUTE RELATIVE GAIN AND RELATIVE ANGLE OF LEFTHAND SIDE-LOBES * UC7SCN45
  4251            90 CONTINUE                                                         UC7SCN46
  4252               GAIN = DV(NXPT) - ALOBGN(1)                                      UC7SCN47
  4253               ANGLE = XORG + (NXPT-1)*PTSPC                                    UC7SCN48
  4254               RELANG=ANGLE - ANGMN
  4255          C                                                                     UC7SCN50
  4256               WRITE(6,89) NLL,GAIN,ANGLE,RELANG
  4257            89 FORMAT(59X, I3, 3(13X,F13.7))
  4258          C                                                                     UC7SCN53
  4259               NLL = NLL+1                                                      UC7SCN54
  4260          C**** EXIT AFTER TWENTY SIDE-LOBES OR DROP THROUGH AT END OF DATA ***** UC7SCN55
  4261               IF( NLL .GT. 20 ) GO TO 95                                       UC7SCN56
  4262            93 CONTINUE                                                         UC7SCN57
  4263          C                                                                     UC7SCN58
  4264               WRITE(6,86)                                                      UC7SCN59
  4265          C                                                                     UC7SCN60
  4266            95 RETURN                                                           UC7SCN61
  4267               END                                                              UC7SCN62
```

SECTION 7

SUPERVISORY MODULES

This section includes those modules that supervise existing stimulus/transfer function modules to simulate the frequency scanned and time scanned array radar systems. This group includes ANTARY and TSRPAT, which are located in Volume III.

SECTION 8

SUBORDINATE MODULES

This section includes all those modules which are subordinate to the larger stimulus/transfer function modules, peripheral modules or supervisory modules. They include both subroutines and functions. Included in this group are the following modules:

ABORT
ANTINT
AZGAIN/ELGAIN
BLOCK DATA
DBLKX
IBOOL
IFLD
IPACK
PACK
RRAND

The following subordinate modules are related to the bistatic target model and target imaging and are located in Volume I, Part 3:

BISTGT
GAM
BESS
EXPI

EXPI is also located in Volume IV, Part 2.

SUBROUTINE ABORT

1. MODULE IDENTIFICATION:

Name	Classification	Reference Number
ABORT	Subordinate	Not user referenced
ERRMSG	Subordinate	Not user referenced

2. PURPOSE:

This subroutine is used to print coded error messages to the user.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NCODE	R	I	Error message code number. $0 < \text{NCODE} \leq 999$

4. CALLING SEQUENCES:

CALL ABORT (NCODE)

A message is printed which states a fatal error has occurred and execution will terminate.

CALL ERRMSG (NCODE)

A message is printed which states a non-fatal error has occurred, a fix-up procedure done, and execution continued.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. Flow Chart: Page 9-81

b. Cross Reference Table: Page 9-218

8-3

```
    2254                SUBROUTINE ABORT(NCODE)
    2255                COMMON/SYS/ MODULE
    2256                WRITE(6,1000) NCODE,MODULE
    2257          1000  FORMAT(' ERROR : ',I3,' OCCURRED DURING EXECUTION OF MODULE : ',
    2258               * I3,'.....FATAL ERROR: JOB WILL TERMINATE' )
    2259                CALL EXIT
    2260                ENTRY ERRMSG(NCODE)
    2261                WRITE(6,1001) NCODE,MODULE



02/11/75       INPUT LISTING                      AUTOFLOW CHART SET - FWC/SCL  RADSIM

  CARD NO         ****                      CONTENTS                               ****

    2262          1001  FORMAT(' ERROR : ',I3,' OCCURRED DURING EXECUTION OF MODULE : ',
    2263               * I3,'.....FIX-UP DONE, REFER TO C.P.D. ; JOB WILL CONTINUE')
    2264                RETURN
    2265                END
```

SUBROUTINE ANTINT

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
ANTINT	Subordinate	(301)

2. PURPOSE:

This subroutine generates an interpolation table from the user provided sampled antenna pattern data. The interpolation table is subsequently used by the functions AZGAIN and ELGAIN to compute antenna gain.

3. INPUT PARAMETERS:

Name	O/R	T	Description
NPT	R	I	Number of user specified antenna gain values
BSIT	O	F	Angle between zero angle of user specified pattern and zero angle in radar coordinates (usually = 0.0)
ANTP(2,75)	R	F	User specified gain pattern array. ANTP(1,J) is the gain of the Jth sample. ANTP(2,J) is the angle of the Jth sample.

4. CALLING SEQUENCES:

CALL ANTINT (NPT, BSIT, ANTP, CDEF)

Where: ANTP contains the Input Antenna Pattern
COEF contains the Output Interpolation coefficients table.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. If no antenna pattern is specified then the coefficients table is set up to represent an omnidirectional antenna of unity gain.

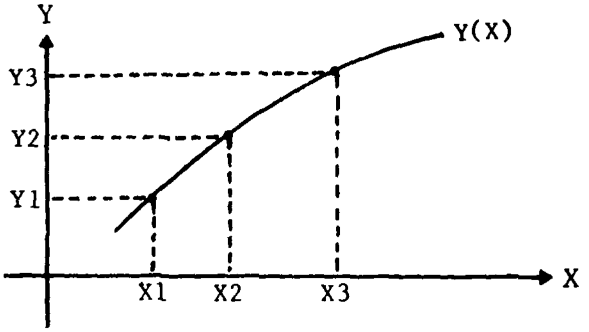
b. Flow Chart: Page 9-140

c. Cross Reference Table: Page 9-226.

6. THEORY OF OPERATION

This module implements a simplified antenna representation which is based on computing the antenna gain as the product of samples from an azimuth and elevation pattern cuts through the peak of the main lobe. The azimuth pattern is true antenna gain in power units. The elevation pattern is normalized to unity gain at the peak of the main lobe.

The parabolic interpolation scheme mechanized herein is described in the following paragraph. Given three points a parabolic curve can be determined which passes through them.



$$Y(X) = X^2 A + XB + C$$

where A, B, and C are the coefficients to be determined

$$A = \frac{XY12 - XY13}{X2 - X3}$$

$$B = XY12 - A * (X1 + X2)$$

$$C = Y1 - (X1A + B) * X1$$

$$XY12 = \frac{Y1 - Y2}{X1 - X2} \quad \text{and } XY13 = \frac{Y1 - Y3}{X1 - X3}$$

In determining the coefficient table elements a double parabolic coefficient is used where possible. This is done by determining 2 sets of coefficients for 2 curves through 4 points and averaging the coefficients. Typically a double parabolic curve fit is smoother than a single parabolic curve fit.

```
      SUBROUTINE ANTINT(NPT,BSIT,ANTP,COEF)
C
      DIMENSION ANTP(2,75),COEF(4,76)
      IF(NPT.GT.2) GO TO 50
      COEF(1,1)=0.0
      COEF(2,1)=0.0
      COEF(3,1)=0.0
      COEF(4,1)=1.0
      COEF(1,2)=360.0
      COEF(2,2)=0.0
      COEF(3,2)=0.0
      COEF(4,2)=0.0
      IF(NPT.NE.2) GO TO 40
      COEF(1,1)=ANTP(2,1)+BSIT
      COEF(4,1)=ANTP(1,1)
      COEF(1,2)=ANTP(2,2)+BSIT
   40 COEF(1,76)=BULL(2)
      NPT=2
      GO TO 40
   50 CONTINUE
      COEF(1,76)=BULL(NPT)
```

```
  3654               NPT2=NPT-2
  3655               A1=0.0
  3656               B1=0.0
  3657               C1=0.0
  3658               FLG =1.0
  3659               DO 100 K=1,NPT2
  3660               L=K+1
  3661               M=K+2
  3662               XY12=(ANTP(1,K)-ANTP(1,L))/(ANTP(2,K)-ANTP(2,L))
  3663               XY13=(ANTP(1,K)-ANTP(1,M))/(ANTP(2,K)-ANTP(2,M))
  3664               A2=(XY12-XY13)/(ANTP(2,L)-ANTP(2,M))
  3665               B2=XY12-A2*(ANTP(2,K)+ANTP(2,L)+2.0*BSIT)
  3666                TEMP=ANTP(2,K)+BSIT
  3667                C2=ANTP(1,K)-TEMP*(A2*TEMP+B2)
  3668               C2=ANTP(1,K)-ANTP(2,K)*(A2*ANTP(2,K)+B2)
  3669         C
  3670               COEF(1,K)=ANTP(2,K)+BSIT
  3671               COEF(2,K)=FLG*(A1+A2)
  3672               COEF(3,K)=FLG*(B1+B2)
  3673               COEF(4,K)=FLG*(C1+C2)
  3674               A1=A2
  3675               B1=B2
  3676               C1=C2
  3677               FLG =0.5
  3678           100 CONTINUE
  3679         C
  3680               COEF(1,L)=ANTP(2,L)+BSIT
  3681               COEF(2,L)=A1
  3682               COEF(3,L)=B1
```

```
      COEF(4,L)=C1
      COEF(1,M)=ANTP(2,M)*BSIT
      COEF(2,M)=0.0
      COEF(3,M)=0.0
      COEF(4,M)=0.0
C
   90 CONTINUE
      WRITE(6,95)NPT,BSIT
   95 FORMAT(1H1,'SUBROUTINE ANTINT     NPT =',I3,'     BSIT =',1PE13.7)
      WRITE(6,105) (( ANTP(J,K),J=1,2),( COEF(L,K),L=1,4),K=1,NPT)
  105 FORMAT (1H ,6(1PE20.7))
C
      RETURN
      END
```

SUBROUTINE AZGAIN/ELGAIN

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
AZGAIN	Subordinate	Not user referenced
ELGAIN	Subordinate	Not user referenced

2. PURPOSE:

This function is to compute the antenna azimuth or elevation gain for any angle within the range of the user specified antenna data.

3. INPUT PARAMETERS:

Name	O/R	T	Description
ANTANG	R	F	The angle of the antenna main lobe in radar coordinates

4. CALLING SEQUENCES:

CALL G = AZGAIN (AZANGL) * ELGAIN (ELANGL)

Where: GA is the antenna gain corresponding to an azimuth angle of AZANGL and elevation angle of ELANGL.

5. RESTRICTIONS, REQUIREMENTS AND MISCELLANEOUS DATA

a. The coefficients table must have been successfully initialized by the subroutine ANTINT before the function can be executed.

b. Flow Charts: Pages 9-159/160

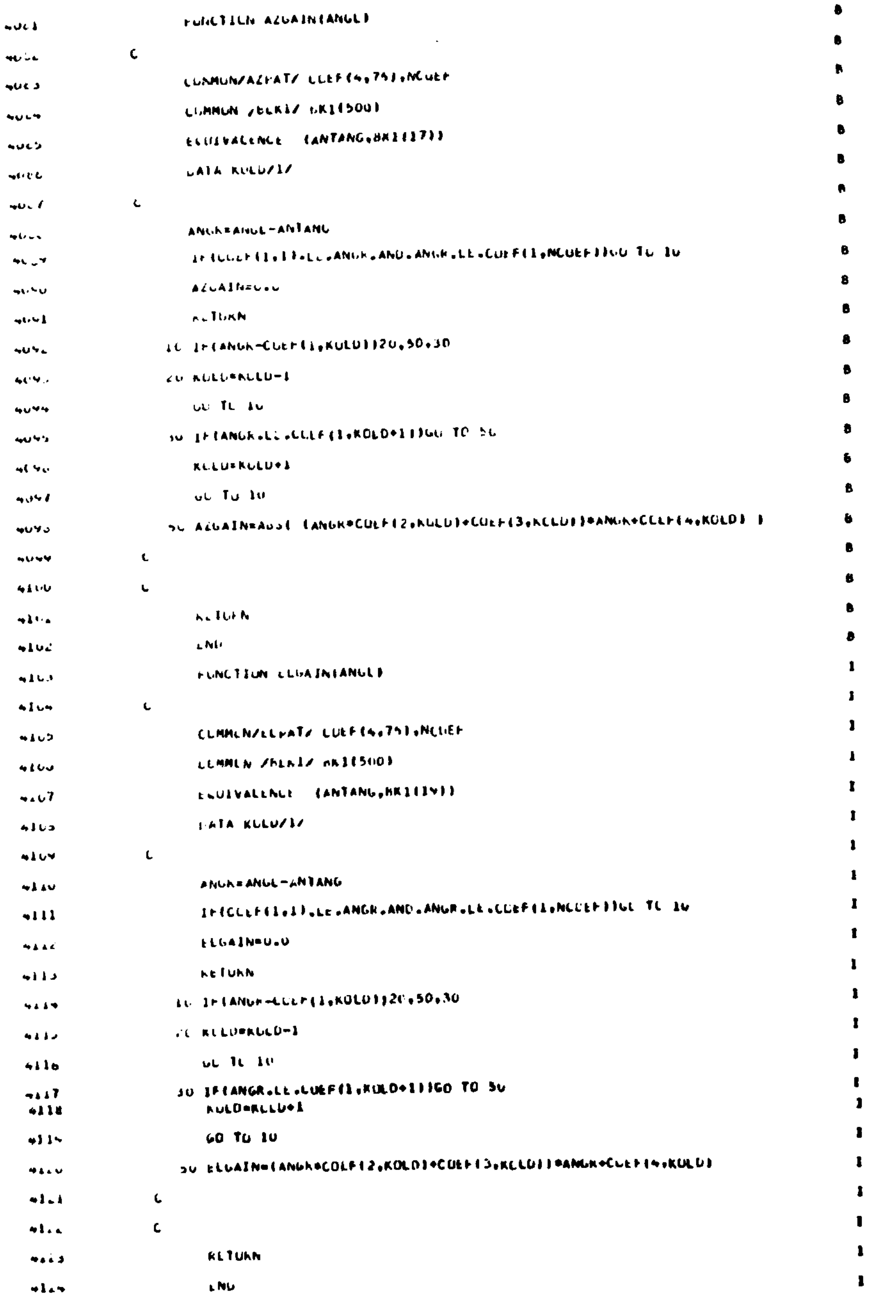
c. Cross Reference Tables: Page 9-229.

6. THEORY OF OPERATION

The input angle ANGL is tested to determine if it is within the range of the input data. If it is outside the range, the gain value returned is zero. Otherwise, the gain is computed from the gain coefficients (A,B, and C) that are closest in angle to ANGL.

AZGAIN = (ANGR ** 2) * A + ANGR * B + C

Note: AZANGL is the angle measured in degrees CCW from any desired reference direction. ELANGL is the angle measured in degrees up from the horizon.



```
      FUNCTION AZGAIN(ANGL)
C
      COMMON/AZPAT/ COEF(4,75),NCOEF
      COMMON /BLK1/ BK1(500)
      EQUIVALENCE  (ANTANG,BK1(17))
      DATA KOLD/1/
C
      ANGR=ANGL-ANTANG
      IF(COEF(1,1).LE.ANGR.AND.ANGR.LE.COEF(1,NCOEF))GO TO 10
      AZGAIN=0.0
      RETURN
   10 IF(ANGR-COEF(1,KOLD))20,50,30
   20 KOLD=KOLD-1
      GO TO 10
   30 IF(ANGR.LE.COEF(1,KOLD+1))GO TO 50
      KOLD=KOLD+1
      GO TO 10
   50 AZGAIN=ABS( (ANGR*COEF(2,KOLD)+COEF(3,KOLD))*ANGR+COEF(4,KOLD) )
C
C
      RETURN
      END
      FUNCTION ELGAIN(ANGL)
C
      COMMON/ELPAT/ COEF(4,75),NCOEF
      COMMON /BLK1/ BK1(500)
      EQUIVALENCE  (ANTANG,BK1(19))
      DATA KOLD/1/
C
      ANGR=ANGL-ANTANG
      IF(COEF(1,1).LE.ANGR.AND.ANGR.LE.COEF(1,NCOEF))GO TO 10
      ELGAIN=0.0
      RETURN
   10 IF(ANGR-COEF(1,KOLD))20,50,30
   20 KOLD=KOLD-1
      GO TO 10
   30 IF(ANGR.LE.COEF(1,KOLD+1))GO TO 50
      KOLD=KOLD+1
      GO TO 10
   50 ELGAIN=(ANGR*COEF(2,KOLD)+COEF(3,KOLD))*ANGR+COEF(4,KOLD)
C
C
      RETURN
      END
```

SUBROUTINE BLOCK DATA

1. *MODULE IDENTIFICATION*:

Name	Classification Code	Reference Number
BLOCK DATA	Subordinate	None

2. PURPOSE:

This non-executable module initializes the labled common BLKRND which contains the random number array used by the random number generator.

3. INPUT PARAMETERS:

None

4. CALLING SEQUENCES:

None

This module is used only at load time to initialize the labeled area.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The numbers used to initialize the random number array were derived from tables of random numbers published by the RAND Corporation: A Million Random Digits with 100,000 Normal Deviates, the Free Press of Glencoe, New York, 1955.

b. Flow Chart: Page 9-124

c. Cross Reference Table: Page 9-224

8-11

```
      BLOCK DATA
      COMMON/BLKRND/ IDMY1,IRND,IADD1,JRND,IDUM(2),NKND1(65),NKND2(64)
      DATA IDMY1,IADD1/1,1/
      DATA NKND1/15154161997,27509664404,30323512272,30068227364,
     *           14051007293,16402190290,26306990212,11260717646,
     *           16201629773,11849273156,19404991345,06977712830,
     *           02863434137,33025570091,11012391622,13431362861,
     *           31267410086,13462139250,20403855902,24219774290,
     *           11557820695,30512809719,12630506319,17722760814,
     *           04722597022,16900280091,16243824041,16328044806,
     *           26212690402,13576024754,11128309920,29134237221,
     *           13164942096,24908960258,03604950630,24513426529,
     *           25262307942,16416251777,32749370959,21116173576,
     *           19292173043,20743061171,21219359579,19674491967,
     *           19244391324,02840123356,27142309994,15625175936,
     *           10410417213,23416520791,22825036692,16006745449,
     *           01762080304,17006962875,16841462774,26473284721,
     *           17160292937,29260744156,8835554466,30699537202,
     *           16000801292,5883873659,14224721680,12081451742,
     *           81604108807/
      DATA NKND2/12052152044,00247664059,15410722660,19302343178,
     *           25022201290,13988647055,01734737402,07229355567,
     *           28120785669,32320962560,19471392797,07603759917,
     *           24780072234,29317499972,07114243643,16232712423,
     *           29170304246,26856574612,20739822612,14601397540,
     *           25672962242,31374670076,13676667951,30403132192,
     *           20172304002,16121261142,14974210487,10223010920,
     *           16210257399,18938138707,01226074097,19062214195,
     *           03577065402,16742167695,11616643767,11174114620,
     *           30292427161,30197487160,22209497006,33362415769,
     *           17239421632,25322444250,26007050175,12422081553,
     *           30224148581,07655423387,32626402591,13101024674,
     *           30533512969,07218771539,00229536870,29196604401,
     *           22112305420,29107810503,10534007415,02097320176,
     *           34914602222,5604776474,30256045180,10652799361,
     *           12174114080,10556707007,10140202992,97790171197/
      END
```

SUBROUTINE DBLKX

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
DBLKX	Subordinate	Not user referenced
SDBLKX	Subordinate	Not user referenced

2. PURPOSE:

This subroutine is used to transfer a defined block of data from one array to another array.

3. INPUT PARAMETERS:

a. DBLKX

Name	O/R	T	Description
NST	R	I	First element of the input array to be transferred. Also, the first element of the output array to be used.
NWORD	R	I	Number of elements to be transferred.

b. SDBLKX

Name	O/R	T	Description
NST	R	I	First element of the input array to be transferred.
NWORD	R	I	Number of elements to be transferred.
LOC	R	I	First element of the output array to be used.

4. CALLING SEQUENCES:

CALL DBLKX (NST, NWORD, X, Y)

Where X contains the Input Waveform
Y contains the Output Waveform

Y(NST+J) = X(NST+J)   J=1, NWORD

CALL SDBLKX (LOC, NST, NWORD, X,Y)

Where: X contains the Input Waveform
          Y contains the Output Waveform

Y(LOC+J) = X(NST+J)   J=1, NWORD

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. All input data to this subroutine is provided through the calling argument list. This subroutine is not directly available to the program user, but is used by other subprograms for data transfer.

b. Flow Chart: Page 9-122

c. Cross Reference Table: Page 9-223.

```
      SUBROUTINE DBLKX(NST,NWORD,X,Y)                          UC7DBL01
      DIMENSION X(1),Y(1)                                      UC7DBL02
      K=NST                                                    UC7DBL03
      GO TO 20                                                 UC7DBL04
      ENTRY SDBLKX(LOC,NST,NWORD,X,Y)                          UC7DBL05
      K=LOC                                                    UC7DBL06
   20 CONTINUE                                                 UC7DBL07
      NSTOP=NST+NWORD-1                                        UC7DBL08
      DO 10 J=NST,NSTOP                                        UC7DBL09
      Y(J)=X(K)                                                UC7DBL10
      K=K+1                                                    UC7DBL11
   10 CONTINUE                                                 UC7DBL12
      RETURN                                                   UC7DBL13
      END
```

FUNCTION IBOOL

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Name
IBOOL	N/A	-

2. PURPOSE:

This function was used when checking out RADSIM on IBM computers. It has been replaced by the standard Honeywell intrinsic function BOOl.

3. INPUT PARAMETERS: N/A

4. CALLING SEQUENCES: N/A

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

   a. Flow Chart: Page 9-119

   b. Cross Reference Table: Page 9-223

```
      SUBROUTINE DBLKX(NST,NWORD,X,Y)                                   UC7DBL01
      DIMENSION X(1),Y(1)                                               UC7DBL02
      K=NST                                                             UC7DBL03
      GO TO 20                                                          UC7DBL04
      ENTRY SDBLKX(LOC,NST,NWORD,X,Y)                                   UC7DBL05
      K=LOC                                                             UC7DBL06
   20 CONTINUE                                                          UC7DBL07
      NSTOP=NST+NWORD-1                                                 UC7DBL08
      DO 10 J=NST,NSTOP                                                 UC7DBL09
      Y(J)=X(K)                                                         UC7DBL10
      K=K+1                                                             UC7DBL11
   10 CONTINUE                                                          UC7DBL12
      RETURN                                                            UC7DBL13
      END
```

FUNCTION IFLD

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
IFLD	N/A	-

2. PURPOSE:

This function was used when checking out RADSIM on IBM computers. It has been replaced by the standard Honeywell intrinsic function FLD.

3. INPUT PARAMETERS: N/A

4. CALLING SEQUENCES: N/A

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

   a. Flow Chart: Page 9-123

   b. Cross Reference Table: Page 9-224

```
                                                                 UC7DBL14
          FUNCTION IFLD(IST,NBITS,IWORD)                         UC7IFD01
          IFLD=FLD(IST,NBITS,IWORD)                              UC7IFD02
          RETURN                                                 UC7IFD03
          END                                                    UC7IFD04
```

FUNCTION IPACK

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
IPACK	Subordinate	Not User Referenced

2. PURPOSE:

This function subprogram is used to pack two or more groups of bits into a standard computer word (36 bits).

3. INPUT PARAMETERS:

Name	O/R	T	Description

ISTR	R	I	The lowest bit into which data transfer is to occur. The bit number is counted starting with the LSB.
			ISTR = 0 corresponds to the LSB and
			ISTR = 35 corresponds to the MSB.

IDATA	R	I	The data which is to be packed into the output word.
IWORD	R	I	The word into which the data is to be packed.

4. CALLING SEQUENCES:

CALL IOUT = IPACK (ISTR, IDATA, IWORD)

IOUT = IWORD ⊕ (IDATA*(2**ISTR))   ⊕ = inclusive or

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA

a. The field (bit locations) of the output word which is to receive the input data must contain zeros before execution of the function.

b. If the data to be placed in the output word is larger than the allocated field then either the data in a higher field will be destroyed or if it is the highest field then some of the most significant bits of IDATA will extend past the standard word length and be lost.

c. Flow Chart: Page 9-118

d. Cross Reference Table: Page 9-223

6. <u>THEORY OF OPERATION</u>

The input data, IDATA, is multiplied by the power of two equivalent to the number of bit positions that it must be shifted in order to line up with its field in the output word. Once the data has been shifted to the proper position, then it is merged into the output word, IWORD, by using an inclusive OR operation.

This function is not directly available to the simulation user, but is used by subprogram within the simulation.

```
      FUNCTION IPACK(ISTP,IDATA,IWORD)                                  UC7IPK01
      ITEMP=IDATA*(2**ISTP)                                             UC7IPK02
      IPACK = BOOL (OR(ITEMP, IWORD) )                                  UC7IPK03
      RETURN                                                            UC7IPK05
      END                                                               UC7IPK06
```

SUBROUTINE PACK

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
PACK	SUBORDINATE	Not user referenced

2. PURPOSE:

This subroutine takes a word in integer format and converts the lower 13 bits of the word into two 8 bit ASCII characters. The ASCII characters are subsequently transmitted to a remote plotting terminal via TSS. Remote plotter accuracy assumed to be 12 bits maximum.

3. INPUT PARAMETERS:

Name	O/R	T	Description
IWD	R	I	Output array word location pointer.
IBIT	R	I	Bit displacement to be used in placing the characters in the output word

4. CALLING SEQUENCES:

CALL PACK (IDAT, IWD, IBIT, IARY, $IIII)

Where: IDAT is the Input word
IARY is the Output array
IIII is the statement number to which control is transferred if the output array is full (68 characters)

5. RESTRICTIONS, RECOMMENDATIONS, MISCELLANEOUS DATA

a. The maximum array length is restricted to 68 characters to be compatible with TSS.

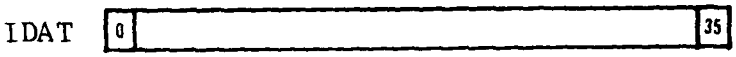
b. The maximum allowable accuracy of the word is 13 bits, i.e. bits 0 through 24 must be zero.

c. Flow Chart: Page 9-88

d. Cross Reference Table: Page 9-219

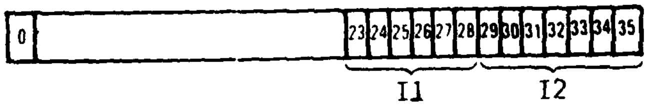
6. THEORY OF OPERATION

The input word IDAT is shown pictorially as follows:



Bit 0 is the MSB and Bit 35 is the LSB.

This word is divided into two characters, I1 and I2as illustrated by the following:



Next I2 is tested to determine if any TSS command characters have occurred. If so, the value of I2 is incremented by +1 or -1 as follows:

TSS Character	Actual No.	Increment
DC1 (XON)	21_8	+1
CAN	30_8	+1
EOT	4_8	+1
RUBOUT	177_8	-1
SOH	1	-1
ETX	3	-1

Since I1 is 6 bits in length a rubout (177_8) cannot occur. Therefore, the remaining TSS characters to protect against have values which are less than 32_{10}. Accordingly, I1 is tested and if its value is less than 32_{10} then bit 7 of I1 is set "on" to preclude occurrence of the undesired character.

Finally, I1 and I2 are packed into the output word pointed to by IWD. The parameter, IBIT, specifies the displacement of I1 from bit 0 of the output word, IARY (IWD).

```
2419          SUBROUTINE PACK(IDAT,IWD,IBIT,IARY,*)
2420          DATA IRUT/O177/,IBIT7/O100/,IXON/O21/,ICANC/O30/
2421           DATA IEOT/4/,IZ/O172/,ICONA/1/,ICONC/3/
2422           DIMENSION IARY(1)
2423           I=IARY(IWD)
2424          I1=FLD(23,6,IDAT)
2425          I2=FLD(29,7,IDAT)
2426           IF(I1.LT.32.AND.I1.GT.0) I1=I1+IBIT7
2427           IF(I2.EQ.IRUT.OR.I2.EQ.ICONA.OR.I2.EQ.ICONC) I2=I2-1
2428          IF(I2.EQ.IXON.OR.I2.EQ.ICANC.OR.I2.EQ.IEOT) I2=I2+1
2429           IF(I2.EQ.IZ) I2=I2+1
2430           FLD(IBIT,9,IARY(IWD))=I1
2431           IBIT=IBIT+9
2432          FLD(IBIT,9,IARY(IWD))=I2
2433           IBIT=IBIT+9
2434           IF(IBIT.LT.35) RETURN
2435           IBIT=0
```

```
09/11/79      INPUT LISTING                    AUTOFLOW CHART SET - FWD/SCL  RADSIM

 CARD NO      ****                         CONTENTS                                  ****

  2436              IWD=IWD+1
  2437              IF(IWD.GT.17) RETURN 1
  24[illegible]              RETURN
  2439              END
```

FUNCTION RRAND

1. MODULE IDENTIFICATION:

Name	Classification Code	Reference Number
RRAND	Subordinate	101, 102 (Parameter initialization only)

2. PURPOSE:

This function generates random numbers for use in various subprograms of the simulation model. Samples from the uniform, Gaussian and Rayleigh distributions can be generated. In addition, two statistical target models are incorporated which are based on the Chi-square distribution with 2 and 4 degrees of freedom.

3. INPUT PARAMETERS: (User supplied data through namelist NL101)

Name	O/R	T	Description
MAD1	0	I	The starting address for selecting random numbers from the random number array (1≤MAD≤128)
UMEAN	0	F	Mean value of the uniform distribution
UUEXT	0	F	Width of the uniform distribution
XMEAN	0	F	Mean value of the Guassian distribution
SIGMA	0	F	Standard deviation of the Guassian and Rayleigh distributions, and average cross section for target models
NTYPE	R	I	Control integer which specifies the type of distribution to be generated

Name	O/R	T	Description
NTYPE	R	I	NTYPE = 1 Uniform distribution (floating point output) = 2 Rayleigh = 3 Gaussian = 4 Uniform distribution (integer 0 to 2^{35}) = 5 Not used = 6 Swerling Target Models #1 and #2 = 7 Swerling Target Models #3 and #4 = 8 Sine
NRAND	R	I	Array containing random numbers which are used to generate the output variates

Variables initialized during execution of the initializer load module based on user supplied data. These intermediate values are generated in order to minimize execution time of the function, i.e., to avoid repeating the same calculations for each entry to RRAND.

```
SIG2SQ = 2.0 *SIGMA * SIGMA
UL     = UMEAN - 0.5 * UUEXT
UEXT   = UUEXT/2 ** 35
```

4. CALLING SEQUENCE:

VAR = RRAND (NTYPE)

VAR will contain the random sample generated by the function from the NTYPE probability distribution.

5. RESTRICTIONS, REQUIREMENTS, MISCELLANEOUS DATA:

a. Before any call can be made to the function RRAND th the input data must be loaded using namelist 101. Any subsequent changes in random variate distributions are also made through namelist 101.

b. For convenience and to minimize program steps the array IRAND was equivalenced to the array NRAND but displaced by one location. This structure allows an address of zero to be used, i.e., an address of zero will access IRAND(0) which overlays NRAND(1\. If this were not done, a test would have to be performed on MAD1 to ensure that an address of zero did not occur.

c. A list of random numbers suitable for initializing the array NRAND are included at the end of this section.

d. Flow Chart Page 9-120

e. Cross Reference Table Page 9-223

6. TH≡ORY OF OPERATION:

For each call to the function RRAND a number IRND is selected from the random number table IRAND. The address of the number selected from the table is MAD1 which is also a random number. The number IRND is added to the random number JRND which was generated by the previous execution of the function. The sign bit is set to zero to ensure a positive number. By adding the two random numbers and truncating the overflow, a new random number is generated which is also called IRND. IRND is placed in the random number table location previously occupied by the original IRND. In this manner the random number table is updated by generating new random numbers and inserting them in the table. From this random number IRND, 7 bits are selected to determine the new address MAD1 to be used in the next call to the function. The 7-bit address field allows the addresses to range from 0 to 127. The random number IRND just generated is an integer having an uniform distribution from 0 to 2^{35} - 1. Once the random number is generated JRND is set equal to IRND for use in the subsequent executions of the function. In order to convert this number to a floating point number r having a uniform distribution from 0 to 1.0, the following conversion is used.

$$r = \text{FLOAT(IRND)}/2^{35}$$

From this uniform distribution other probability distributions can be generated by using transformations which map a uniform distribution into the desired distribution. The following is a list of the transformations used in the function·

a. Uniform distribution $a \leq x \leq b$

$$x_n = (b - a)r_n + a$$

b. Rayleigh distribution $P(x) = \frac{x}{\sigma^2} e^{-\left(\frac{x^2}{2\sigma^2}\right)}$

$$x_n = \sqrt{-2\sigma^2 \ln r_n}$$

c. Gaussian distribution $P(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\left(\frac{x^2}{2\sigma^2}\right)}$

$$x_n = \sqrt{2\sigma^2 \ln r_n} * \mathrm{COS}\ 2\pi r_{n+1}$$

d. Swerling Target Models #1 & #2 $P(x) = \frac{1}{\bar{\sigma}} e^{-\left(\frac{\sigma}{\bar{\sigma}}\right)}$

$x_n = -\bar{\sigma} \ln r_n$ $\bar{\sigma}$ = average cross section

This distribution characterizes a target with a large number of independent scatterers of approximately equal cross section (Swerling cases 1 and 2).

e. Swerling Target Models #3 & #4 $P(x) = \frac{4\sigma}{\bar{\sigma}^2} e^{-\left(\frac{2\sigma}{\bar{\sigma}}\right)}$

$x_{2n} = \frac{-\bar{\sigma}}{2} (\ln r_n + \ln r_{n+1})$ = average cross section

This distribution characterizes a target with one dominant scatterer plus smaller scatterers or one large reflector subject to small changes in orientation (Swerling Cases 3 and 4).

8-25

```
      FUNCTION RRAND(NTYPE)                                         UC7RNY01
      COMMON/CCRRND/IDMY1, IRND, MAD1, JRND, UMEAN, UUEXT, XMEAN,   UC7RNY03
     *SIGMA,UUM,SIG2SQ,UL,UEXT,NRAND(129)                           UC7RNY04
      DIMENSION IRAND(128)                                          UC7RNY05
      DATA IREPT/0/,IMULT/1220703125/
       DATA IMAX/4294967296/,N2P16/65536/
      DATA C1/2.9103830E-11/                                        UC7RNY07
      EQUIVALENCE (IRAND(1),NRAND(2))                               UC7RNY08
   10 KRND=IRAND(MAD1)                                              UC7RNY09
      IRND=KRND*JRND*IMULT                                               210
      IRND = FLD(1,35,IRND)                                         UC7RNY11
      IRAND(MAD1)=IRND                                              UC7RNY12
      MAD1 = FLD(15,7,IRND)                                         UC7RNY13
       IF(IDMY1.EQ.1) WRITE(6,1000) IRND,IRND,MAD1,MAD1,KRND,KRND,
     * JRND,JRND
 1000  FORMAT(1H , 4(I15,O15))
      GO TO (200,300,400,500,600,700,800,900),NTYPE                      250
  200 RRAND = FLOAT(IRND)*UEXT + UL                                 UC7RNY15
      JRND = IRND                                                   UC7RNY16
      RETURN                                                        UC7RNY17
  300 RRAND = SQRT(SIG2SQ*ALOG(FLOAT(IRND )*C1))                    UC7RNY18
      JRND = IRND                                                   UC7RNY19
      RETURN                                                        UC7RNY20
  900 CONTINUE                                                           320
  400 I1=FLD(1,17,KRND)-N2P16
      I2=FLD(16,17,KRND)-N2P16
      IS=I1*I1+I2*I2                                        350
      IF(IS.LT.IMAX) GOTO 20
      JRND=IRND
      GOTO 10
   20 S=1.0/FLOAT(IS)
      VCOS=S*FLOAT(I1*I1-I2*I2)                             380
      VSINE=S*2.0*FLOAT(I1*I2)                              390
      IF(NTYPE.EQ.3) GO TO 405                              400
      KRAND=VCOS                                            410
      DUM=VSINE                                             420
      JRND=IRND                                             430
      RETURN                                                440
  405 DUM=SQRT(SIG2SC*ALOG(FLOAT(JRND)*CI))                 480
      KRAND= DUM*VCOS + XMEAN                          UC7RNY38
      DUM = DUM*VSINE + XMEAN                          UC7RNY39
      JRND = IRND                                      UC7RNY40
      RETURN                                           UC7RNY41
  500 KRAND = GOUL(IRND)                               UC7RNY42
      JRND=IRND                                        UC7RNY43
      RETURN                                           UC7RNY44
  600 KRAND=DUM                                        UC7RNY45
      RETURN                                           UC7RNY46
  700 KRAND=-SIGMA*ALOG(FLOAT(IRND)*CI)                UC7RNY47
      JRND=IRND                                        UC7RNY48
      RETURN                                           UC7RNY49
  800 IF(IREPT.EQ.1) GO TO 801                         UC7RNY50
      SUM=0.0                                          UC7RNY51
  801 SUM=SUM+ALOG(FLOAT(IRND)*CI)                     UC7RNY52
      IREPT=IREPT+1                                    UC7RNY53
      IF(IREPT.NE.2) GO TO 10                          UC7RNY54
      IREPT=0                                          UC7RNY55
      KRAND=-SIGMA*SUM/2.0                             UC7RNY56
      JRND=IRND                                        UC7RNY57
      RETURN                                           UC7RNY58
      END                                              UC7RNY59
```

LIST OF RANDOM NUMBERS

The following list of random numbers is suitable for initializing the random number array NRAND used by the function subprogram RRAND. The numbers were derived from tables of random numbers published by RAND Corporation: A Million Random Digits with 100,000 Normal Deviates, the Free Press of Glencoe, New York, 1955.

```
*NL101   NRAND= 12068158044,06847664659,15416782760,19382343178,
33122308420,25052201640,13968647055,01734737406,07289355507,
16534467415,24386072834,29317493972,07114843643,16232716423,
06160416880,16410917613,23416520791,28825638452,10800745449,
 29107616508,23120765669,32320902560,15471392797,07683759917,
 03669736170,29170504246,26866574818,20335880812,14861357546,
 03491463822,25072568248,31374670078,13676667951,30463132192,
 05804776974,20172084006,16184261842,14974210467,10283018420,
 30256545185,18310257399,18938188207,01286074697,19662214195,
 10832795361,01577045480,16742867695,11666646767,18174114680,
 18174114680,30892487160,30892487160,28360949700,3368415709,
 10556707007,17235921632,25322444850,30007056175,13488881553,
 10140208896,30224146581,07659423387,32626402591,13101024674,
 09779017119,30533512969,07218771539,00229536870,29198604401,
 01702686304,15134181997,27509664464,30323512272,30068227398,
 17006458873,14051007893,16402190290,26306990212,11260717646,
 16841462774,16801629773,11849273156,19404991345,06977712830,
 26473264721,02883434137,33025570091,11012391622,13431365861,
 17160292937,31267410036,13462139250,26463685902,24219774296,
 29620744156,11557820695,30512809719,12630506319,17722780814,
 08883554436,04722597022,16900280091,16243624041,16388044606,
 03669953728,26212698408,13570004754,11168309528,29134237821,
 16068801392,13164942096,29908968258,03564986686,24513426529,
 05883873859,25262307992,16416251777,32749370939,21116176576,
 14824731680,19395173043,20743061171,21319359579,19074491967,
 18081451743,19244390324,08846123356,27142309994,15825176938,
```

SECTION 9

RADSIM COMPUTER PROGRAM

FLOW CHARTS

PUBLISHED IN PART 2 OF THIS VOLUME.

SECTION 10

INDEX

(See also Section 6 in Volume I, Part 1
and Appendix B in Volume IV, Part 1)

PROGRAM/ SUBROUTINE/ FUNCTION	MODULE NUMBERS	DESCRIPTION & LISTING PAGE	FLOW CHART PAGE	CROSS REF. PAGE
ECM	512	4-45	9-104	9-221
ECMFL	-	See DIGFIL		
ECMFSU	-	See DIGFSF		
ELGAIN	-	8-8	9-160	9-229
ERGYCP	118	6-16	9-70	9-217
ERGYRE	116/117	See ERGYCP		
ERRMSG	-	See ABORT		
FGENMP	421	See FGENXY		
FGENXY	420	4-51	9-72	9-217
FILT	407/408	4-60	9-55	9-215
FWDET	416/417	See HWDET		
HET	-	4-64	9-103	9-221
HLIM	438/439	See HWDET		
HWDET	414/415	4-65	9-137	9-225
IBOOL	-	8-14	9-119	9-223
IFLD	-	8-15	9-123	9-224
IFT	-	See ZFFT		
IFWDET	447/448	See HWDET		
IHLIM	442/443	See HWDET		
IHWDET	445/446	See HWDET		
INGTOR	409/410	4-70	9-97	9-220
IONOS	511	4-74	9-117	9-223
IPACK	-	8-16	9-118	9-223
ISQDET	449/450	See HWDET		
LAMPCP	458	4-80	9-68	9-217
LAMPRE	456/457	See LAMPCP		
MAIN 1	-	2-1	9-2	9-206
MAIN 2	-	2-2	9-34	9-210
MTIFLT	430/431	4-83	9-201	9-234
MTIIFT	-	See MTIFLT		
MTIINC	-	See MTIFLT		
MTINCL	432/433	See MTIFLT		
NCSWPI	436/437	See SWPINT		
NONLIN	401/402	4-91	9-135	9-225
OUTCUM	210/211	See CUMDIS		
PACK	-	8-18	9-88	9-219
PDF	212/213	See CUMDIS		

PROGRAM/ SUBROUTINE/ FUNCTION	MODULE NUMBERS	DESCRIPTION & LISTING PAGE	FLOW CHART PAGE	CROSS REF. PAGE
PHDEC	508/509	4-94	9-78	9-218
PHENC	506/507	4-100	9-167	9-229
PLOTTR	307-310	6-18	9-90	9-219
PLTFMT	113	6-23	9-175	9-231
PTLIST	303-306	6-28	9-82	9-218
PXFRM	454	Vol III	9-147	9-227
RDIGFL	405	4-105	9-100	9-221
RECF	451	Vol III	9-145	9-227
RECFTF	452	See RECF		
RNDARY	214/215	4-109	9-150	9-227
RRAND	101/102	8-21	9-120	9-223
RSHFTS	231/232	See SHIFT		
RSHIFT	229/230	See SHIFT		
RTOPDB	103	6-35	9-156	9-228
RTOPM	110	See RTOPDB		
RTOPM2	111	See RTOPDB		
SCANNR	313	6-40	9-161	9-229
SDBLKX	-	See DBLKX		
SHIFT	224/225	4-113	9-63	9-216
SHIFTS	226/227	See SHIFT		
SPCAVG	312	Vol III	9-165	9-229
SQDET	418/419	See HWDET		
SWPINT	434/435	4-118	9-204	9-234
TARGET	501	4-122	9-109	9-222
TGTNCL	502	See TARGET		
TSARM	-	See TSARY		
TSARY	426	Vol III	9-186	9-232
TSARY 1	427	See TSARY		
TSARY 2	-	See TSARY		
TSRPAT	505	Vol III	9-178	9-231
WEITCP	222	See WEITRE		
WEITMP	223	See WEITRE		
WEITRE	221	4-129	9-57	9-215
WVGUID	510	4-135	9-116	9-223
XYTODB	104/108	See RTOPDB		
XYTOM	105	See RTOPDB		
XYTOM2	106	See RTOPDB		

PROGRAM/ SUBROUTINE FUNCTION	MODULE NUMBERS	DESCRIPTION & LISTING PAGE	FLOW CHART PAGE	CROSS REF. PAGE
ZFFT	202	5-7	9-125	9-224
ZIFFT	203	See ZFFT		

SECTION 11

RECURSIVE COMPUTATION OF SINE/COSINE PAIRS

Many computer programs require the evaluation of sine and cosine functions for angles that are uniformly spaced over the interval from 0^o to 360^o, for example, DFT subroutine. In many cases the CPU time required to execute these computer programs can be significantly reduced by the use of the following digital oscillator to generate sine/cosine pairs. The Z-plane representation of the digital oscillator is shown in Figure 11-1.

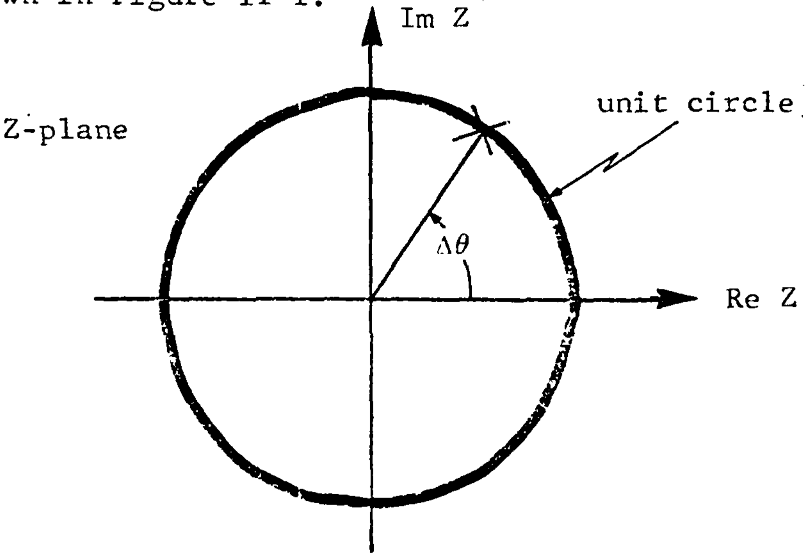


Figure 11-1 Z-PLANE REPRESENTATION OF THE DIGITAL OSCILLATOR

The variable $\Delta\theta$ is the angular increment for which the sine/cosine pairs are to be calculated. The digital oscillator block diagram is shown in Figure 11-2.

The stimulus[$\cos\theta_o + j \sin\theta_o$]is applied only when n=0 and is zero otherwise. Before execution of the digital oscillator routine, the storage registers (A & B) used to generate the unit delays must be cleared.

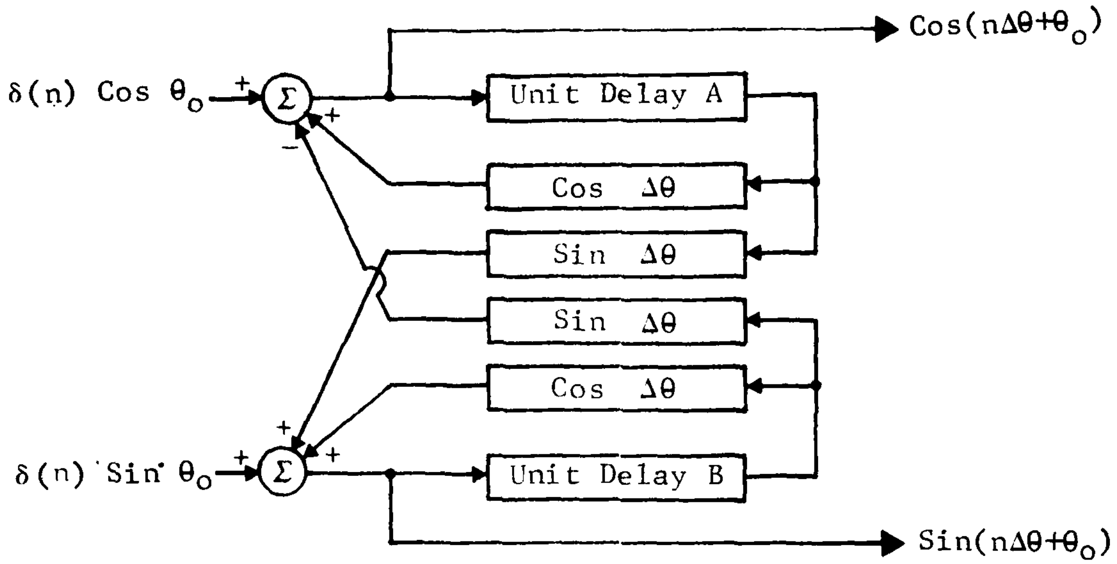


Figure 11-2 DIGITAL OSCILLATOR BLOCK DIAGRAM

n = number of executions of the digital oscillator routine.

The Fortran statements for implementing the digital oscillator are the following:

```
                A = Cos (θo)
                B = Sin (θo)
                DELC = Cos (Δθ)
                DELS = Sin (Δθ)
      |-------->
      |             {A and B contain the
      |             {sine/cosine of   [θo+(n-1)Δθ]}
    Loop
    n=1, N      TEMP = A
      |         A = A * DELC - B * DELS
      |-------- B = TEMP * DELS + B * DELC
```

where $\Delta\theta$ = angle increment.
θ_o = starting angle.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

AMERICAN REVOLUTION BICENTENNIAL
1776-1976